PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

#2

In re the Application of

Taro TERAO

Application No.: New U.S. Patent Application

Filed: June 6, 2000 Docket No.: 106408

For: ONE-WAY FUNCTION GENERATION METHOD, ONE-WAY FUNCTION VALUE GENERATION DEVICE, PROVING DEVICE, AUTHENTICATION METHOD, AND AUTHENTICATION DEVICE

CLAIM FOR PRIORITY

Director of the U.S. Patent and Trademark Office
Washington, D.C. 20231

Sir:

The benefit of the filing date of the following prior foreign application filed in the following foreign country is hereby requested for the above-identified patent application and the priority provided in 35 U.S.C. §119 is hereby claimed:

Japanese Patent Application No. 11-244519 filed on August 31, 1999.

In support of this claim, a certified copy of said original foreign application:

__x__ is filed herewith.

_____ was filed on _____ in Parent Application No. _____ filed _____.

It is requested that the file of this application be marked to indicate that the requirements of 35 U.S.C. §119 have been fulfilled and that the Patent and Trademark Office kindly acknowledge receipt of this document.

Respectfully submitted,

James A. Oliff
Registration No. 27,075

JAO:TJP/crt

Thomas J. Pardini
Registration No. 30,411

Date: June 6, 2000

DEPOSIT ACCOUNT USE AUTHORIZATION
Please grant any extension necessary for entry;
Charge any fee due to our Deposit Account No. 15-0461

OLIFF & BERRIDGE, PLC
P.O. Box 19928
Alexandria, Virginia 22320
Telephone: (703) 836-6400

日　本　国　特　許　庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。
This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: １９９９年　８月３１日

出願番号
Application Number: 平成１１年特許願第２４４５１９号

出願人
Applicant (s): 富士ゼロックス株式会社

２０００年　４月２１日

特許庁長官
Commissioner,
Patent Office 近藤隆彦

出証番号　出証特２０００－３０２８０１４

【書類名】　特許願

【整理番号】　FN99-00214

【提出日】　平成11年 8月31日

【あて先】　特許庁長官殿

【国際特許分類】　H04L　9/32

【発明の名称】　一方向性関数生成方法、一方向性関数値生成装置、証明装置、認証方法および認証装置

【請求項の数】　38

【発明者】

【住所又は居所】　神奈川県足柄上郡中井町境４３０　グリーンテクなかい　富士ゼロックス株式会社内

【氏名】　寺尾　太郎

【特許出願人】

【識別番号】　000005496

【氏名又は名称】　富士ゼロックス株式会社

【電話番号】　0462-38-8516

【代理人】

【識別番号】　100086531

【弁理士】

【氏名又は名称】　澤田　俊夫

【電話番号】　03-5541-7577

【手数料の表示】

【予納台帳番号】　038818

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　一方向性関数生成方法，一方向性関数値生成装置，証明装置，認証方法および認証装置

【特許請求の範囲】

【請求項１】　一方向性関数Ｈと固有値ｄに依存した一方向性関数Ｘを生成する生成方法であって，関数生成固有値ｓを保持し，関数生成固有値ｓと固有値ｄより値生成固有値ｕを作成し，値生成固有値ｕとメッセージＭより一方向性関数Ｈを用いてメッセージＭの一方向性関数値Ｘ（Ｍ）を作成する一方向性関数生成方法。

【請求項２】　請求項１に記載の一方向性関数生成方法であって，関数生成固有値ｓと固有値ｄより一方向性関数Ｇを用いて値生成固有値ｕが計算されることを特徴とする一方向性関数生成方法。

【請求項３】　請求項１に記載の一方向性関数生成方法であって，関数生成固有値ｓと固有値ｄより慣用暗号の暗号化関数Ｅを用いて値生成固有値ｕが計算されることを特徴とする一方向性関数生成方法。

【請求項４】　請求項１に記載の一方向性関数生成方法であって，値生成固有値ｕとメッセージＭより一方向性関数Ｈと慣用暗号の暗号化関数Ｄを用いてメッセージＭの一方向性関数値Ｘ（Ｍ）が計算されることを特徴とする一方向性関数生成方法。

【請求項５】　固有値ｄに依存する一方向性関数Ｘを計算する一方向性関数値生成装置であって，固有値ｄを入力する手段と，メッセージＭを入力する手段と，関数生成固有値ｓを保持する手段と，関数生成固有値ｓと固有値ｄより値生成固有値ｕを作成する手段と，値生成固有値ｕとメッセージＭより一方向性関数Ｈを用いてメッセージＭの一方向性関数値Ｘ（Ｍ）を作成する手段とを有する一方向性関数値生成装置。

【請求項６】　請求項５に記載の一方向性関数値生成装置であって，値生成固有値ｕおよび一方向性関数値Ｘ（Ｍ）の計算過程を外部から観測することが困難なことを特徴とする一方向性関数値生成装置。

【請求項７】　メッセージＭに依存する秘密鍵による処理を行う証明装置で

あって，メッセージＭを入力する手段と，値生成固有値ｕを保持する手段と，値生成固有値ｕとメッセージＭより一方向性関数Ｈを用いてメッセージＭの一方向性関数値Ｘ（Ｍ）を作成する手段と，秘密鍵Ｘ（Ｍ）による処理を行う手段とを有し，値生成固有値ｕは，関数生成固有値ｓと固有値ｄより作成されることを特徴とする証明装置。

【請求項８】　請求項７に記載の証明装置であって，値生成固有値ｕおよび秘密鍵Ｘ（Ｍ）による処理の計算過程を外部から観測することが困難なことを特徴とする証明装置。

【請求項９】　請求項７に記載の証明装置であって，スマートカードなどの携帯可能な小型演算装置として構成されることを特徴とする証明装置。

【請求項１０】　請求項７に記載の証明装置であって，ＣＰＵ内部のモジュールとして構成されることを特徴とする証明装置。

【請求項１１】　請求項７に記載の証明装置であって，秘密鍵による処理を行う手段が，チャレンジｃを入力する手段と，チャレンジｃと秘密鍵Ｘ（Ｍ）からレスポンスｒを計算する手段と，レスポンスｒを出力する手段からなることを特徴とする証明装置。

【請求項１２】　請求項７に記載の証明装置であって，秘密鍵による処理を行う手段が，チャレンジｃを入力する手段と，乱数ｋを生成する手段と，乱数ｋとチャレンジｃと秘密鍵Ｘ（Ｍ）からレスポンスｒを計算する手段と，レスポンスｒを出力する手段からなることを特徴とする証明装置。

【請求項１３】　請求項７に記載の証明装置であって，秘密鍵による処理を行う手段が，乱数ｋを生成する手段と，乱数ｋからコミットメントｗを計算する手段と，コミットメントｗを出力する手段と，チャレンジｃを入力する手段と，乱数ｋとチャレンジｃと秘密鍵Ｘ（Ｍ）からレスポンスｒを計算する手段と，レスポンスｒを出力する手段からなることを特徴とする証明装置。

【請求項１４】　請求項７に記載の証明装置であって，秘密鍵による処理を行う手段が，乱数ｋを生成する手段と，乱数ｋからコミットメントｗを計算する手段と，コミットメントｗを出力する手段と，チャレンジｃを入力する手段と，乱数ｋとコミットメントｗとチャレンジｃと秘密鍵Ｘ（Ｍ）からレスポンスｒを

計算する手段と，レスポンスｒを出力する手段とを有することを特徴とする証明装置。

【請求項１５】　請求項７に記載の証明装置であって，秘密鍵による処理を行う手段が，有限体上の乗法群の乗法およびべき乗演算を行うことを特徴とする証明装置。

【請求項１６】　請求項７に記載の証明装置であって，秘密鍵による処理を行う手段が，有限体上の楕円曲線の加法およびスカラー倍演算を行うことを特徴とする証明装置。

【請求項１７】　請求項７に記載の証明装置であって，秘密鍵による処理を行う手段が，ｎを因数分解が困難な合成数としてとして法ｎのもとで乗算剰余およびべき乗剰余演算を行うことを特徴とする証明装置。

【請求項１８】　請求項７に記載の証明装置であって，メッセージＭは利用条件を含み，メッセージを入力する手段は，メッセージＭに含まれた利用条件が満たされない場合はメッセージの入力を拒絶することを特徴とする証明装置。

【請求項１９】　請求項７に記載の証明装置であって，メッセージＭは秘密鍵処理のパラメターを含み，秘密鍵による処理を行う手段は，メッセージＭに含まれた秘密鍵処理のパラメターに基づいて処理を行うことを特徴とする証明装置。

【請求項２０】　固有値ｄに応じて証明器Ｔを発行する装置であって，固有値ｄを入力する手段と，関数生成固有値ｓを保持する手段と，関数生成固有値ｓと固有値ｄより値生成固有値ｕを作成する手段と，値生成固有値ｕを証明器Ｔに書き込む手段とを有し，証明器Ｔは，値生成固有値ｕを保持しており，メッセージＭの入力に対して値生成固有値ｕとメッセージＭより一方向性関数Ｈを用いてメッセージＭの一方向性関数値Ｘ（Ｍ）を作成して秘密鍵Ｘ（Ｍ）による処理を行うことを特徴とする証明器発行装置。

【請求項２１】　資格発行者が資格付与者にメッセージＭに対応づけて資格を発行し，資格検証者が資格付与者の資格を検証する認証方法であって，資格発行者は，関数生成固有値ｓと資格付与者に対応する固有値ｄより値生成固有値ｕを作成し，値生成固有値ｕとメッセージＭより一方向性関数Ｈを用いてメッセー

ジMの一方向性関数値Ｘ（Ｍ）を計算し，秘密鍵Ｘ（Ｍ）と対をなす公開鍵ｙを証明する証明証Ｃを資格付与者に発行し，資格付与者は，証明証Ｃを資格検証者に提示し，値生成固有値ｕとメッセージMより一方向性関数Ｈを用いてメッセージMの一方向性関数値Ｘ（Ｍ）を計算し，秘密鍵Ｘ（Ｍ）による処理を行い，資格検証者は，証明証Ｃを検証し，また，資格付与者の秘密鍵Ｘ（Ｍ）による処理を証明証Ｃの証明する公開鍵ｙで検証することによって認証を行う認証方法。

【請求項２２】　請求項２１に記載の認証方法であって，資格発行者の発行する証明証Ｃには認証の種別を表す識別子ａｉｄが記載され，資格検証者は，証明証Ｃに記載された認証識別子ａｉｄが，行おうとする認証の種別に一致するときに限って，証明証Ｃの検証に成功することを特徴とする認証方法。

【請求項２３】　請求項２１に記載の認証方法であって，資格発行者の発行する証明証Ｃには利用条件が記載され，資格検証者は，証明証Ｃに記載された利用条件が満たされるときに限って，証明証Ｃの検証に成功することを特徴とする認証方法。

【請求項２４】　固有値ｄとメッセージMに応じて証明証Ｃを発行する証明証発行装置であって，固有値ｄを入力する手段と，メッセージMを入力する手段と，関数生成固有値ｓを保持する手段と，関数生成固有値ｓと固有値ｄより値生成固有値ｕを作成する手段と，値生成固有値ｕとメッセージMより一方向性関数Ｈを用いてメッセージMの一方向性関数値Ｘ（Ｍ）を作成する手段と，秘密鍵Ｘ（Ｍ）と対を成す公開鍵ｙを作成する手段と，公開鍵ｙを証明する証明証Ｃを発行する手段とからなる証明証発行装置。

【請求項２５】　メッセージMに応じて認証を行う認証装置であって，メッセージMを入力する手段と，値生成固有値ｕを保持する手段と，値生成固有値ｕとメッセージMより一方向性関数Ｈを用いてメッセージMの一方向性関数値Ｘ（Ｍ）を作成する手段と，秘密鍵Ｘ（Ｍ）による処理を行う手段と，秘密鍵Ｘ（Ｍ）と対をなす公開鍵ｙを証明する証明証Ｃを保持する手段と，証明証Ｃを検証する手段と，公開鍵ｙで秘密鍵による処理を検証する手段とからなり，値生成固有値ｕは，関数生成固有値ｓと固有値ｄより作成されることを特徴とする認証装置。

【請求項２６】　資格発行者が資格付与者にメッセージMに対応づけて資格を発行し，資格検証者が資格付与者の資格を検証する認証方法であって，資格発行者は，関数生成固有値ｓと資格付与者に対応する固有値ｄより値生成固有値ｕを作成し，値生成固有値ｕとメッセージMより一方向性関数Ｈを用いてメッセージMの一方向性関数値Ｘ（Ｍ）を計算し，秘密鍵ｘと一方向性関数値Ｘ（Ｍ）とから定まるアクセスチケットｔを資格付与者に発行し，資格付与者は，値生成固有値ｕとメッセージMより一方向性関数Ｈを用いてメッセージMの一方向性関数値Ｘ（Ｍ）を計算し，秘密鍵Ｘ（Ｍ）による処理を行い，秘密鍵Ｘ（Ｍ）による処理をアクセスチケットｔによって秘密鍵ｘによる処理に変換し，資格検証者は，資格付与者の秘密鍵ｘによる処理を秘密鍵ｘと対をなす公開鍵ｙで検証することによって認証を行う認証方法。

【請求項２７】　請求項２１あるいは請求項２６に記載の認証方法であって，メッセージMには認証の種別を表す識別子ａｉｄが含まれることを特徴とする認証方法。

【請求項２８】　固有値ｄとメッセージMに応じてアクセスチケットｔを発行するアクセスチケット発行装置であって，固有値ｄを入力する手段と，メッセージMを入力する手段と，関数生成固有値ｓを保持する手段と，関数生成固有値ｓと固有値ｄより値生成固有値ｕを作成する手段と，値生成固有値ｕとメッセージMより一方向性関数Ｈを用いてメッセージMの一方向性関数値Ｘ（Ｍ）を作成する手段と，秘密鍵ｘと一方向性関数値Ｘ（Ｍ）よりアクセスチケットｔを作成する手段と，アクセスチケットｔを発行する手段とを有するアクセスチケット発行装置。

【請求項２９】　請求項２８に記載のアクセスチケット発行装置であって，アクセスチケットｔが，秘密鍵ｘと一方向性関数値Ｘ（Ｍ）の差ｘ－Ｘ（Ｍ）として計算されることを特徴とするアクセスチケット発行装置。

【請求項３０】　請求項２８に記載のアクセスチケット発行装置であって，アクセスチケットｔが，秘密鍵ｘと一方向性関数値Ｘ（Ｍ）の商ｘ／Ｘ（Ｍ）として計算されることを特徴とするアクセスチケット発行装置。

【請求項３１】　値生成固有値ｕを（u_1，．．．，u_m）とし，一方向性関

数値Ｘ（Ｍ）を一方向性関数Ｈの値のビット連結Ｈ（u_1｜Ｍ）｜．．．｜Ｈ（u_m｜）から生成して所望のビット長とする請求項２８記載のアクセスチケット生成装置。

【請求項３２】　値生成固有値（u_1，．．．，u_m）を，関数生成固有値ｓを（s_1，．．，s_m）として一方向性関数Ｇを用いてu_j＝Ｇ（s_j｜ｄ）から求める請求項３１記載のアクセスチケット生成装置。

【請求項３３】　メッセージＭに応じて認証を行う認証装置であって，メッセージＭを入力する手段と，値生成固有値ｕを保持する手段と，値生成固有値ｕとメッセージＭより一方向性関数Ｈを用いてメッセージＭの一方向性関数値Ｘ（Ｍ）を作成する手段と，秘密鍵Ｘ（Ｍ）による処理を行う手段と，秘密鍵ｘと一方向性関数値Ｘ（Ｍ）とから定まるアクセスチケットｔを保持する手段と，秘密鍵Ｘ（Ｍ）による処理をアクセスチケットｔによって秘密鍵ｘによる処理に変換する手段と，秘密鍵ｘと対をなす公開鍵ｙを保持する手段と，公開鍵ｙで秘密鍵による処理を検証する手段とを有し，値生成固有値ｕは，関数生成固有値ｓと固有値ｄより作成されることを特徴とする認証装置。

【請求項３４】　請求項３３に記載の認証装置であって，秘密鍵による処理を変換する手段が，チャレンジｃをアクセスチケットｔによって更新する手段からなることを特徴とする認証装置。

【請求項３５】　請求項３３に記載の認証装置であって，秘密鍵による処理を変換する手段が，レスポンスｒをアクセスチケットｔによって更新する手段からなることを特徴とする認証装置。

【請求項３６】　請求項３３に記載の認証装置であって，秘密鍵による処理を変換する手段が，レスポンスｒをアクセスチケットｔとチャレンジｃによって更新する手段からなることを特徴とする認証装置。

【請求項３７】　請求項３３に記載の認証装置であって，秘密鍵による処理を変換する手段が，チャレンジｃをコミットメントｗによって更新する手段と，レスポンスｒをアクセスチケットｔとチャレンジｃによって更新する手段からなることを特徴とする認証装置。

【請求項３８】　請求項３３に記載の認証装置であって，秘密鍵による処理

を変換する手段が，チャレンジｃをアクセスチケットｔとコミットメントｗによって更新する手段と，レスポンスｒをコミットメントｗによって更新する手段からなることを特徴とする認証装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は，情報セキュリティ技術に関するものであり，特に，一方向性関数を生成する方法および一方向性関数値を生成する装置とそれらによる認証の方法および装置に関するものである。

【０００２】

【従来の技術】

[従来技術の概要]

認証を行うための秘密鍵を，スマートカードなどの耐タンパー性を有する容器に封入して配布することは広く行われている。耐タンパー性容器に封入された秘密鍵を用いた次のような認証システムを構成することができる。

【０００３】

説明の簡略化のため，秘密鍵の配布者をセンター，秘密鍵を格納する耐タンパー性容器をトークン，秘密鍵の受領者をユーザーと呼称する。

【０００４】

センターは，秘密鍵ｘを生成し，トークンに封入し，トークンをユーザーに配布する。トークンは，また，秘密鍵ｘに基づいた処理を実行可能なように構成する。例えば，以下の様にすれば良い。

【０００５】

センターは公開鍵暗号系を定める。例えば，Ｇを離散対数問題が困難な有限アーベル群（記法の簡略化のために加法的に記すが，もちろん，慣習的に乗法的に記される群であっても，以下の議論は単に記法を変更するのみで適用できる），ｐを素数，F_pをｐ元体，ｇ：$F_p \rightarrow G$を体F_pの加法群から有限群Ｇへの自明でない準同型，ｙ：$F_p \rightarrow G$をｙ＝ｇｘ（ここでｘは体F_pの加法群のｘ倍自己準同型と同一視）で定義される準同型，$\pi : G \rightarrow F_p$を写像とする。このような有限

群Ｇとして，ある有限体上の乗法群や楕円曲線を取ることができることは良く知られている。

【０００６】

トークンは，入出力手段，乱数生成手段，写像πｇの計算手段と有限素体F_pの算法の実行手段を持ち，入力されたチャレンジｃに対して，乱数ｋを生成して，レスポンスｒ＝（r_0，r_1）を

【０００７】

【数１】

$$r_0 = c - \pi(g(k))$$

$$r_1 = k - r_0 x$$

と計算して，出力する。

【０００８】

一方，検証器は，入出力手段，乱数生成手段，準同型ｇの計算手段，準同型ｙの計算手段，有限群Ｇの算法の実行手段，有限素体F_pの算法の実行手段とを持ち，チャレンジｃを乱数として生成して出力し，入力されたレスポンスｒに対して，

【０００９】

【数２】

$$c = r_0 + \pi(g(r_1) + y(r_0))$$

が成り立つことを検証する。

【００１０】

以上の検証器とトークンを組み合わせることによって，検証器の送信したチャレンジｃに対して，秘密鍵ｘによるチャレンジｃのＮｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名をレスポンスｒとして返信することで認証を行うことができる。

【００１１】

この手法は，例えば，アクセス制御の対象となるアプリケーションプログラムに検証器の機構を組み込み，そのアプリケーションプログラムの利用資格としてトークンを配布し，プログラムの実行時の要所要所で，上記の質疑応答による認証を行うことで，アクセス制御に応用することができる。アプリケーションプロ

グラム中の認証を行うコードは，攻撃者がこれらを削除する事を防ぐために，容易に発見されないようにしなければならない。

【００１２】

上に説明した手法を直截に適用すると，例えば，同様な手法でアクセス制御された独立なアプリケーションプログラムを複数利用する場合，必要な認証の種類が増加し，認証の種類の数だけのトークンが必要になるが，以下に説明するように，一つのトークンのみで複数の認証に対応することもできる。

【００１３】

この例では，センターは各トークン毎に異なる秘密鍵ｘを封入して，ユーザーに配布する。また，各アプリケーションプログラムには認証識別子ａｉｄが割り当てられている。アプリケーションプログラムの提供者がユーザーに認証識別子ａｉｄに対応したアプリケーションプログラムの利用権を付与する場合，認証識別子ａｉｄとユーザーの所持するトークンの公開鍵ｙとを含み，アプリケーションプログラムの提供者の署名が施された証明証Ｃをユーザーに発行する。

【００１４】

アプリケーションプログラムは，プログラムの実行時，ユーザーの利用資格を認証するタイミングで，証明証Ｃを読み込んで署名を検証し，記載された認証識別子ａｉｄが自身の認証識別子に相等しいことを確認し，記載されたトークンの公開鍵ｙに基づいた認証を前述のように実行する。

【００１５】

アクセス資格認証のケーパビリティとしてトークンに格納された秘密鍵に対応する公開鍵を含む証明証を用いる方法は，秘密鍵をトークンに封入して配布するセンターが信用できる第三者と仮定すると，複数の互いに利害関係を持たないアプリケーションプログラムの提供者が共通に利用することができ，またユーザーも単一のトークンを保持するだけで済むので優れている。また，例えば，トークンのモジュールをＣＰＵに予め内蔵する構成をとれば，センターがユーザーにトークンを配布する手間をなくし，ユーザーもトークンを意識することなく利用することができる。

【００１６】

一方，このような方法は，トークンの公開鍵が大域的なユーザーの識別子となり，ケーパビリティが大規模に収集された場合，トークンの公開鍵で名寄せが行われるなど，ユーザーのプライバシーの観点からは望ましくない。

【００１７】

一方，耐タンパー性容器に秘密ハッシュ関数を封入して配布することによって，上の問題点を解消する次のような認証システムを構成することができる。

【００１８】

この場合，センターは，秘密ハッシュ関数Ｘを生成し，トークンに封入し，トークンをユーザーに配布する。ここで，秘密ハッシュ関数Ｘは各トークン毎に異なるものとする。

【００１９】

アプリケーションプログラムの提供者がユーザーに利用資格を表すケーパビリティを前述したように証明証Ｃの形で付与する場合，例えば，認証識別子ａｉｄの秘密ハッシュ関数Ｘによるハッシュ値Ｘ（ａｉｄ）に対応するトークンの公開鍵ｙ＝ｇ（Ｘ（ａｉｄ））を証明証Ｃに含ませる。

【００２０】

今回は認証識別子ａｉｄ毎に対応する公開鍵ｙが異なったものになるので，ケーパビリティを収集しても，トークンの公開鍵による名寄せは不可能になる。

【００２１】

利用資格の認証を行う場合には，トークンに認証を行う認証識別子ａｉｄが入力され，これに対応する秘密鍵ｘ＝Ｘ（ａｉｄ）が生成されるステップが増えるのみで後は同様である。

【００２２】

また，トークンがトークン毎に異なる秘密ハッシュ関数Ｘを持つものとして，特開平１０－２４７９０５号公報のアクセス資格認証装置では，以下に説明するような，上に説明したものとは異なるケーパビリティが提案されている。

【００２３】

ここでは，ｙ＝ｇ（ｘ）を満たす公開鍵ｙと秘密鍵ｘの対は，各ユーザーが所持するトークン内での秘密ハッシュ値Ｘ（ａｉｄ）とは独立に認証識別子ａｉｄ

にのみ依存するように定められ（例えば，認証識別子ａｉｄを公開鍵ｙ自身として良い），ユーザーに対して発行するケーパビリティをｔ＝ｘ－Ｘ（ａｉｄ）とする。このように秘密鍵ｘとトークン内での秘密ハッシュ値Ｘ（ａｉｄ）から計算される量であるケーパビリティをアクセスチケットと呼ぶことにする。

【００２４】

検証器は，自分の識別子ａｉｄに対応する公開鍵ｙを保持し，前述の質疑応答による認証を行う。

【００２５】

また，ユーザーは，トークンにａｉｄを入力して，トークン内で秘密ハッシュ値Ｘ（ａｉｄ）を生成し，チャレンジｃに対して，秘密鍵Ｘ（ａｉｄ）を用いてレスポンスｒ＝（r_0，r_1）を取得する。

【００２６】

ユーザーは，さらに，トークンの出力したレスポンスｒをアクセスチケットｔによって，

【００２７】

【数３】

$$r_1 \leftarrow r_1 + r_0 t$$

として更新した後に，レスポンスｒを検証器に返信する。

【００２８】

更新されたレスポンスｒは，チャレンジｃに対する秘密鍵ｘによるＮｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名であることは容易に確認できる。

【００２９】

アクセスチケットによる認証方式では，アプリケーションプログラムはケーパビリティを読み込んだり，ケーパビリティの真正性を検証することを省略でき，検証器側（つまり，アプリケーションプログラムのアクセス制御コード）の構造はさらに簡易化され効率が良い。

【００３０】

また，アクセスチケットによる認証方式は，トークン内で生成される秘密ハッシュ値とは独立な秘密鍵による処理を，秘密鍵を開示することなく実行可能とす

るので，単に，質疑応答型の認証に限らず，秘密鍵による処理（例えば，公開鍵暗号における暗号文の復号やメッセージへの署名生成）を行う機能を安全に委託することを可能にするという著しい特徴を備えている。

【００３１】

以上で述べてきた証明証やアクセスチケットによるアクセス制御の方式は，もちろん，アプリケーションプログラムの実行制御のみならず，例えば，ファイルへのアクセス制御やｈｔｔｐサーバーへのアクセス制御等にも適用することが可能である。

【００３２】

また，トークンを可搬性を持ったスマートカードのような形態で提供し，ケーパビリティをトークンに格納することによって，これらのデジタル情報であるケーパビリティで通常のチケットやカードを模倣することも可能である。

【００３３】

このように，トークンに秘密鍵の代わりに秘密ハッシュ関数を封入することにより，多彩な認証方式が利用できるようになる。

【００３４】

[従来技術の問題点]

ハッシュ関数は，通常，暗号プロトコルを構成するプリミティブとしては公開される固定的な関数として利用されるケースが多く，暗号用のＩＣチップにおいても，ＳＨＡ－１（ＳＨＡ：Ｓｅｃｕｒｅ　Ｈａｓｈ　Ａｌｇｏｒｉｔｈｍ）やＭＤ５（ＭＤ：Ｍｅｓｓａｇｅ　Ｄｉｇｅｓｔ）といった標準的なハッシュ関数のみを備えている場合が多いので，トークンの実装コストを考慮すると，これらの標準的なハッシュ関数を流用して，トークン毎に異なる秘密ハッシュ関数を実現する方法を吟味する必要がある。

【００３５】

一つの簡単な方法は，トークンは標準的なハッシュ関数Ｈとトークン毎に異なる秘密固有値ｕを保持し，秘密ハッシュ関数ＸをＸ（Ｍ）＝Ｈ（ｕ｜Ｍ）として実現することである。

【００３６】

この場合，センターはトークン毎に乱数としてｕを生成し，トークンの識別子ｔｉｄとトークン秘密固有値ｕとのタプル（ｔｉｄ，ｕ）をデータベースに保持して置くことが考えられるが，トークンの個数が増加した場合，データ量が膨大となり，また，このデータベースのエントリーは，上記認証システムを構成する上で厳重に守秘されることが必須であるので管理が困難である。

【００３７】

例えば，証明証型のケーパビリティを用いる場合，一つのトークンの秘密ハッシュ関数Ｘが漏洩し，ケーパビリティの発行者がその漏洩の事実に気付かない限り，以後，このトークンと認証識別子ａｉｄに対応する証明証Ｃを発行させるたびに，証明証Ｃとそれに対応する秘密鍵Ｘ（Ｍ）を知る者はトークンを用いることなく検証にパスできるようになる。ただし，この場合は秘密ハッシュ値Ｘ（Ｍ）から漏洩者を追跡することができる。

【００３８】

アクセスチケット型のケーパビリティを用いる場合，一つのトークンの秘密ハッシュ関数Ｘが漏洩し，ケーパビリティの発行者がその漏洩の事実に気付かない限り，以後，このトークンと認証識別子ａｉｄに対応するアクセスチケットｔを発行させるたびに，ｘ＝ｔ＋Ｘ（Ｍ）として認証識別子ａｉｄに対応した秘密鍵ｘが組織的に暴露され深刻な問題となる。しかも，この場合は，暴露された秘密鍵ｘ自身からは漏洩者を追跡できない。

【００３９】

センターが厳重に秘密保持するデータ量を削減するために，センター秘密固有値ｓを導入し，トークン毎の秘密固有値を持たない構成を考える。ここでは，秘密ハッシュ関数ＸはＸ（Ｍ）＝Ｈ（ｓ｜ｔｉｄ｜Ｍ）として実現される。

【００４０】

このような構成では，センターは唯一の秘密情報ｓのみを安全に管理すれば，各トークン毎の秘密固有情報にあたるものを保持しておく必要が無いが，トークンには大域的な秘密情報ｓが封入されることになり，万一，一つのトークンで耐タンパー性の仮定が破られた場合，トークン識別子ｔｉｄを入手可能なデータと考えると全てのトークンの秘密ハッシュ関数が一斉に暴露されてしまい危険が大

きい。

【００４１】

また，今まで述べてきた認証方式は，例として挙げた離散対数問題が困難な有限群Ｇ上のＮｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名だけでなく，様々な公開鍵暗号系の上で構成できる。

【００４２】

まず，離散対数問題の困難性に立脚したＤｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有，ＤＳＡ署名（Ｄｉｇｉｔａｌ　Ｓｉｇｎａｔｕｒｅ　Ａｌｇｏｒｉｔｈｍ），Ｓｃｈｎｏｒｒ認証等の公開鍵暗号系には全て同様に適用できる。

【００４３】

また，零化域決定問題の困難性に立脚したＲＳＡ，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証等の公開鍵暗号系にも適用できる。

【００４４】

この場合，λを有限アーベル群Ｇの全ての元γを零化（つまりλγ＝０）する最小の非零整数とし，有限群Ｇにおいてλの決定が困難とする。このような有限群Ｇとして，有理整数環ＺのＲＳＡ法数ｎによる剰余類環の乗法群（Ｚ／ｎＺ）*を取ることができることは良く知られている。

【００４５】

例えば，公開鍵暗号系として有限群Ｇ上のＧｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名を採用し，秘密ハッシュ値Ｘ（ａｉｄ）を秘密鍵として公開鍵ｙ＝ｐ^{X}（ａｉｄ）を証明する証明証Ｃをケーパビリティとして利用することもできるし，また，秘密鍵ｘと公開鍵ｙ＝ｐ^{x}に対応するアクセスチケットｔ＝ｘ－Ｘ（ａｉｄ）をケーパビリティとして利用することもできる。

【００４６】

また，公開鍵暗号系として有限群Ｇ上のＲＳＡを選び，公開鍵ｙと秘密鍵ｘ＝ｙ$^{-1}$に対応するアクセスチケットｔ＝ｘ－Ｘ（ａｉｄ）をケーパビリティとして利用することもできる。

【００４７】

これらの諸例では，秘密ハッシュ値Ｘ（ａｉｄ）は公開鍵暗号系の秘密値が属

する有限代数系の元とみなされている。秘密値が属する代数系は，Ｎｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名等の離散対数問題系の例では，有限素体F_pであり，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名やＲＳＡ等の零化域決定問題系の例では，それぞれ，有限群Ｇ自体とその忠実な作用環Ｚ／λＺである。

【００４８】

離散対数問題系の例では，現在，推奨されている秘密鍵サイズ（１６０ビット程度）は，通常使用されているハッシュ関数の値のサイズと同程度であるが，零化域決定問題系の例では，推奨されている秘密鍵サイズ（１０２４ビット程度）は，通常使用されるハッシュ関数の値のサイズよりも大きい。

【００４９】

【発明が解決しようとする課題】

以上に説明してきたように，本発明は，センターの管理コストが低く，また，秘密ハッシュ関数を封入されたトークンが万一開封された場合も，秘密情報の暴露が小規模に押さえられ，また，標準のハッシュ関数を流用することで，トークンの開発コストが低く，また，利用する公開鍵暗号の秘密鍵の鍵サイズと同等のサイズの値を有する，秘密ハッシュ関数を配布する方式を提案することを目的としている。

【００５０】

このような秘密ハッシュ関数の配布のためには，標準のハッシュ関数をあるパラメータによって変形したハッシュ関数の族を組織的に生成するハッシュ関数生成方式であって，しかも，その方式で生成されたハッシュ関数の計算において，ハッシュ関数の生成に用いたパラメータ自身を利用しないものがあれば良い。

【００５１】

【課題を解決するための手段】

本発明のハッシュ関数生成スキームは，既知のハッシュ関数Ｈと固有値ｄに依存したハッシュ関数Ｘを生成するにあたって，センターがハッシュ関数生成固有値ｓを保持し，ユーザーには，ハッシュ関数生成固有値ｓと固有値ｄより計算されたハッシュ値生成固有値ｕを封入したハッシュ値生成装置を配布するものであり，ハッシュ値生成固有値ｕとメッセージＭよりハッシュ関数Ｈを用いてメッセ

ージMのハッシュ値X（M）を生成するものである。

【００５２】

本発明おいては、センターのハッシュ関数生成固有値とトークン（ユーザ）の固有値からトークンごとに異なるハッシュ値生成固有値ｕを生成するのでトークンごとに秘密固有値を保持管理する必要がない。また、トークンに保持されているハッシュ値生成固有値ｕが漏洩しても組織的な秘密情報（センターのハッシュ関数生成固有値ｓ）が暴露されるおそれがない。

【００５３】

本発明は特許請求の範囲に記載のとおり一方向性関数生成方法、一方向性関数値生成装置、証明装置、証明器発行装置、認証方法、認証装置、アクセスチケット発行装置として実現することができる。また本発明の少なくとも一部をソフトウェア製品として実現することもできる。

【００５４】

【発明の実施の形態】

まず最初に，以後の説明で共通に使われる記号を説明する。

【００５５】

Ｚは有理整数環，ｐは素数，F_pはｐ元体，｛０，１｝*はビット列のモノイド（算法はビット列の連結であり｜で表す）である。

【００５６】

Ｇは有限アーベル群，$\pi : G \rightarrow F_p$と$\varepsilon : G \rightarrow \{0, 1\}^*$と$\eta : \{0, 1\}^* \rightarrow F_p$は写像とする。

【００５７】

記述を簡略化するために有限群Ｇを加法的に記す。また，γを有限群Ｇの元，νを有理整数としたとき，γのν倍を必ず左作用の形でν．γと書くことにする。有限群Ｇが乗法的に記されている場合は０を１に，γ＋γ’をγγ’に，－を／に，ν．γを

【００５８】

【数４】

γ^{ν}

に置き換えれば良い。

【００５９】

また，ｇは位数がｐの有限群Ｇの元，λは有限群Ｇの零化域

【００６０】

【数５】

$$\mathrm{Ann}\ G=\{\nu\in Z\ ;\ (\forall\gamma\in G)\ \nu.\ \gamma=0\}$$

の生成元とし，Λは有理整数環Ｚの法λによる剰余類環Ｚ／λＺとする。有限環Λは有限群Ｇの忠実な作用環（つまりΛの元αによるＧの元γのスカラー倍がきちんと定義され，Λの元αがＧの任意の元γについてα．γ＝０ならばα＝０）である。

【００６１】

有限群Ｇの与えられた元ｙ＝ｘ．ｇからｐ元体F_pの元ｘを求めることを底ｇについての離散対数問題，有理整数λを求めることを零化域決定問題とよぶことにしよう。以後の実施例で現れる有限群Ｇは，離散対数問題が困難であるか，零化域決定問題が困難であるかのいずれかである。

【００６２】

離散対数問題が困難な有限群Ｇとしては，ｑをある素数の冪として有限体Ｆｑ上の乗法群GL_1ないし楕円曲線Ｅが良く知られている。また，零化域決定問題が困難な有限群Ｇとしては，有理整数環Ｚの合成数ｎによる剰余類環の乗法群が良く知られている。

【００６３】

写像π：Ｇ→F_pとして，ハッシュ関数，法ｐによる剰余，また，ｑ＜ｐとして有限群Ｇが有限体Ｆｑ上の楕円曲線の場合は座標関数（つまり，Ｐ＝（x_P，y_P）に対して，π（Ｐ）＝x_Pとする）等がある。

【００６４】

写像ε：Ｇ→｛０，１｝*として，有限群Ｇの元の自然な符号化（つまり，ビット列としての表現）を対応させる関数やハッシュ関数等がある。

【００６５】

写像η：｛０，１｝*→F_pとして，ハッシュ関数等がある。

【００６６】

［実施例１：ハッシュ値生成装置］

本実施例では，固有値ｄとメッセージＭを入力すると固有値に依存したハッシュ関数Ｘによるメッセージのハッシュ値Ｘ（Ｍ）を出力するハッシュ値生成装置を図１に基づいて説明する。図１はハッシュ値生成装置の構成図である。

【００６７】

固有値入力部１は，ハッシュ関数Ｘの生成に要するパラメータである固有値ｄが入力される。メッセージ入力部２は，ハッシュ値を求めたいメッセージＭが入力される。関数生成固有値記憶部３は，値生成固有値の生成に要するパラメータである関数生成固有値ｓを保持する。値生成固有値算出部４は，関数生成固有値記憶部３に保持された関数生成固有値ｓと固有値入力部１に入力された固有値ｄより値生成固有値ｕを生成する。ハッシュ値算出部５は，値生成固有値算出部４が生成した値生成固有値ｕとメッセージ入力部２に入力されたメッセージＭよりハッシュ関数Ｈを用いてハッシュ値Ｘ（Ｍ）を生成する。ハッシュ値出力部６は，ハッシュ値算出部５が生成したハッシュ値Ｘ（Ｍ）を出力する。

【００６８】

ハッシュ関数Ｈとして，例えば，ＲＳＡ Ｄａｔａ Ｓｅｃｕｒｉｔｙ，Ｉｎｃ．のＭＤ５（ハッシュ値長は１２８ビット）や米国Ｎａｔｉｏｎａｌ Ｉｎｓｔｉｔｕｔｅ ｏｆ Ｓｔａｎｄａｒｄｓ ａｎｄ ＴｅｃｈｎｏｌｏｇｙのＦＩＰＳ１８０－１で定義されたＳＨＡ－１（ハッシュ値長は１６０ビット）を用いて良い。

【００６９】

以下，値生成固有値ｕおよびハッシュ値Ｘ（Ｍ）の具体的な生成方法を，様々なバリエーションを交えて説明しよう。

【００７０】

値生成固有値算出部４は，値生成固有値ｕを，例えば，ハッシュ関数Ｇを用いて，ｕ＝Ｇ（ｓ｜ｄ）として計算しても良い。ここでも，ハッシュ関数ＧとしてＭＤ５やＳＨＡ－１を取ることができる。また，ハッシュ関数Ｇがハッシュ関数Ｈと同一でも良い。

【００７１】

また，例えば，慣用暗号の暗号化関数Ｅを用いて，ｕ＝Ｅ（ｄ，ｓ）として計算しても良い。ここで，固有値ｄを暗号化鍵とすると，値生成固有値ｕのサイズは関数生成固有値ｓのサイズと同じになる。

【００７２】

ハッシュ値算出部５は，ハッシュ値Ｘ（Ｍ）を，例えば，Ｘ（Ｍ）＝Ｈ（ｕ｜Ｍ）として計算しても良い。このようにしてハッシュ値Ｘ（Ｍ）を計算すると，ハッシュ値Ｘ（Ｍ）の長さはハッシュ関数Ｈのハッシュ値長と同じになる。

【００７３】

ハッシュ値Ｘ（Ｍ）の長さをハッシュ関数Ｈのハッシュ値長より大きくする方法を説明しよう。

【００７４】

値生成固有値算出部４は，固有値ｄと関数生成固有値ｓ＝（ｓ１，．．．，ｓｍ）に対して，値生成固有値ｕ＝（u_1，．．．，u_m）を，例えば，ハッシュ関数Ｇを用いて，u_i＝Ｇ（s_i｜ｄ）と計算する。あるいは，固有値ｄをｄ＝（d_1，．．．，d_m）として，u_i＝Ｇ（s_i｜d_i）として計算しても良い。

【００７５】

ハッシュ値算出部５は，ハッシュ値Ｘ（Ｍ）をＸ（Ｍ）＝Ｈ（u_1｜Ｍ）｜．．．｜Ｈ（u_m｜Ｍ）として計算する。このようにするとハッシュ値Ｘ（Ｍ）の長さはハッシュ関数Ｈのハッシュ値長のｍ倍となり，例えば，ハッシュ関数Ｈとして１２８ビットのハッシュ値を持つＭＤ５を取り，ｍ＝８とするとハッシュ値Ｘ（Ｍ）は１０２４ビットとなる。あるいは，ハッシュ関数Ｈとして１６０ビットのハッシュ値を持つＳＨＡ－１を取り，ｍ＝７とし，ハッシュ値Ｘ（Ｍ）をＨ（u_1｜Ｍ）｜．．．｜Ｈ（u_m｜Ｍ）の上位１０２４ビットとしても，１０２４ビットのハッシュ値Ｘ（Ｍ）を得ることができる。

【００７６】

また，例えば，値生成固有値ｕとメッセージＭよりハッシュ関数Ｈと慣用暗号の暗号化関数Ｅを用いてハッシュ値Ｘ（Ｍ）をＸ（Ｍ）＝Ｅ（Ｈ（ｕ｜Ｍ），ｕ）として計算する。ここでは，Ｅの最初の変数が暗号化の鍵とすると，ハッシュ

値Ｘ（Ｍ）の長さは値生成固有値ｕの長さと等しくなる。値生成固有値ｕの長さが大きくなるようにしておけば，このようにしても長いハッシュ値を得られる。

【００７７】

［実施例２：証明装置（確定型２方向認証）］

本実施例では，メッセージＭを入力するとメッセージＭに依存した秘密鍵Ｘ（Ｍ）による処理を行う証明装置を図２と図３に基づいて説明する。図２は証明装置の構成図である。本実施例の証明装置は，ハッシュ値生成装置（実施例１）で用いたものと等価なメッセージ入力部２とハッシュ値算出部５を持つ。図３は秘密鍵処理部８の詳細な構成図である。

【００７８】

値生成固有値記憶部７は，ハッシュ値Ｘ（Ｍ）の生成に要するパラメータである値生成固有値ｕを保持する。ハッシュ値算出部５は，値生成固有値記憶部７に保持された値生成固有値ｕとメッセージ入力部２に入力されたメッセージＭよりハッシュ関数Ｈを用いてハッシュ値Ｘ（Ｍ）を生成する。秘密鍵処理部８は，ハッシュ値算出部５の生成したハッシュ値Ｘ（Ｍ）を秘密鍵として処理を行う。

【００７９】

この証明装置は，例えば，スマートカード等の携帯性を備えたデバイスとして実現しても良い。このようにすると証明装置を常時持ち運び，いろいろな局面で利用することができる。

【００８０】

また，この証明装置は，例えば，ＣＰＵの内部モジュールとして実現してもよい。証明装置を主に計算機の利用時のアクセス資格証明に用いるのであれば，ＣＰＵに予め内蔵された証明装置は，計算機とは別途に入手する必要がなく，また，この証明装置を用いたアクセス資格認証によって，コンテンツを保護したいと考えるコンテンツ提供者側から見ても，証明装置の頒布の手間が省けるので好都合である。

【００８１】

メッセージ入力部２は，入力されるメッセージＭに基づいた処理を行ってもよい。

【００８２】

例えば，メッセージMが有効期限情報等の利用条件を含み，利用条件が満たされない場合は，メッセージの入力を拒絶するように構成してもよい。このように構成することで，秘密鍵による処理の実行に有効期限情報等の利用条件を対応づけることが可能となる。

【００８３】

また，メッセージMが秘密鍵処理のパラメータＧ，ｐ，π，ε，η，ｇ等を含んでいてもよい。このように構成することで，メッセージMに対応づけて秘密鍵処理のパラメータを変更することが可能となる。

【００８４】

また，メッセージMが秘密鍵処理のアルゴリズムの識別子を含んでいてもよい。このように構成することで，メッセージMに対応づけて秘密鍵処理のアルゴリズムを変更することが可能となる。

【００８５】

秘密鍵処理部８は，図３に示すように、チャレンジ入力部９とレスポンス生成部１０とレスポンス出力部１１より構成される。

【００８６】

チャレンジ入力部９は，認証のための質疑情報であるチャレンジｃが入力される。レスポンス生成部１０は，チャレンジ入力部９に入力されたチャレンジｃとハッシュ値算出部６が生成したハッシュ値Ｘ（Ｍ）よりレスポンスｒを生成する。レスポンス出力部１１は，レスポンス生成部１０が生成したレスポンスｒを出力する。

【００８７】

レスポンスｒの生成方法をＤｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有とＲＳＡを例にとって説明しよう。

【００８８】

有限群Ｇは，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有の例では離散対数問題が困難とし，ＲＳＡの例では零化域決定問題が困難とする。ハッシュ値算出部６が生成するハッシュ値Ｘ（Ｍ）は，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有の例ではｐ

元体F_pの元であり，ＲＳＡの例では有限環Λの元である。チャレンジ入力部９に入力されるチャレンジｃは有限群Ｇの元である。

【００８９】

[Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有の場合]

レスポンスｒは

【００９０】

【数６】

$$r = X(M) . c$$

として生成される。

【００９１】

[ＲＳＡの場合]

レスポンスｒは

【００９２】

【数７】

$$r = X(M) . c$$

として生成される。

【００９３】

[実施例３：証明装置（ランダム型２方向認証）]

本実施例では，実施例２と異なる秘密鍵処理部８を持つ証明装置の例を図４に基づいて説明している。図４は秘密鍵処理部８の詳細な構成図である。

【００９４】

秘密鍵処理部８は，乱数生成部１２とチャレンジ入力部９とレスポンス生成部１０とレスポンス出力部１１より構成される。

【００９５】

乱数生成部１２は，乱数ｋを生成する。レスポンス生成部１０は，乱数生成部１２の生成した乱数ｋとチャレンジ入力部９に入力されたチャレンジｃとハッシュ値算出部６が生成したハッシュ値Ｘ（Ｍ）よりレスポンスｒを生成する。

【００９６】

以下，レスポンスｒの具体的な生成方法を，様々なバリエーションを交えて説

明しよう。

【００９７】

有限群Ｇは，ＤＳＡ署名，ＥｌＧａｍａｌ署名の変種，Ｎｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名，Ｓｃｈｎｏｒｒ署名の例では離散対数問題が困難とし，メッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の例では零化域決定問題が困難とする。

【００９８】

ハッシュ値算出部６が生成するハッシュ値Ｘ（Ｍ）と乱数生成部１２が生成する乱数ｋは，ＤＳＡ署名，ＥｌＧａｍａｌ署名の変種，Ｎｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名，Ｓｃｈｎｏｒｒ署名の例ではｐ元体F_pの元であり，メッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の例では有限群Ｇの元である。

【００９９】

チャレンジ入力部９に入力されるチャレンジｃは，ＤＳＡ署名，ＥｌＧａｍａｌ署名の変種，Ｎｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名，メッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の例では，ｐ元体F_pの元であり，Ｓｃｈｎｏｒｒ署名，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の例では任意のビット列である。

【０１００】

［ＤＳＡ署名の場合］

レスポンスｒは

【０１０１】

【数８】

$$r=(r_0,\ r_1)$$

$$r_0=\pi(k.\ g)$$

$$r_1=(c+r_0X(M))/k$$

として生成される。

【０１０２】

［変形ＥｌＧａｍａｌ署名１の場合］

レスポンスｒは

【０１０３】

【数９】

$$r=(r_0,\ r_1)$$

$$r_0=\pi\ (k.\ g)$$

$$r_1=ck-r_0X\ (M)$$

として生成される。

【０１０４】

[変形Ｅ１Ｇａｍａ１署名２の場合]

レスポンスｒは

【０１０５】

【数１０】

$$r=(r_0,\ r_1)$$

$$r_0=\pi\ (k.\ g)$$

$$r_1=r_0k-cX\ (M)$$

として生成される。

【０１０６】

[変形Ｅ１Ｇａｍａ１署名３の場合]

レスポンスｒは

【０１０７】

【数１１】

$$r=(r_0,\ r_1)$$

$$r_0=\pi\ (k.\ g)$$

$$r_1=(r_0+cX\ (M))\ /\ k$$

として生成される。

【０１０８】

[変形Ｅ１Ｇａｍａ１署名４の場合]

レスポンスｒは

【０１０９】

【数１２】

$$r=(r_0,\ r_1)$$

$$r_0=\pi\ (k.\ g)$$

$$r_1=(ck-r_0)\ /X\ (M)$$

として生成される。

【０１１０】

[変形Ｅｌ Ｇａｍａｌ署名５の場合]

レスポンスｒは

【０１１１】

【数１３】

$$r=(r_0,\ r_1)$$

$$r_0=\pi\ (k.\ g)$$

$$r_1=(r_0k-c)\ /X\ (M)$$

として生成される。

【０１１２】

[Ｎｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名の場合]

レスポンスｒは

【０１１３】

【数１４】

$$r=(r_0,\ r_1)$$

$$r_0=c-\pi\ (k.\ g)$$

$$r_1=k-r_0X\ (M)$$

として生成される。

【０１１４】

[Ｓｃｈｎｏｒｒ署名の場合]

レスポンスｒは

【０１１５】

【数１５】

$$r=(r_0,\ r_1)$$

$r_0 = \eta\ (c \mid \varepsilon\ (k.\ g))$

$r_1 = k + r_0 X\ (M)$

として生成される。

【０１１６】

[メッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の場合]

レスポンスｒは

【０１１７】

【数１６】

$r = (r_0,\ r_1)$

$r_0 = c - \pi\ (p.\ k)$

$r_1 = k - r_0.\ X\ (M)$

として生成される。

【０１１８】

[Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の場合]

レスポンスｒは

【０１１９】

【数１７】

$r = (r_0,\ r_1)$

$r_0 = \eta\ (c \mid \varepsilon\ (p.\ k))$

$r_1 = k + r_0.\ X\ (M)$

として生成される。

【０１２０】

[実施例４：証明装置（３方向認証）]

本実施例では，実施例２ないし３と異なる秘密鍵処理部８を持つ証明装置の例を図５に基づいて説明している。図５は秘密鍵処理部８の詳細な構成図である。

【０１２１】

秘密鍵処理部８は，乱数生成部１２とコミットメント生成部１３とコミットメント出力部１３とチャレンジ入力部９とレスポンス生成部１０とレスポンス出力部１１より構成される。

【０１２２】

乱数生成部１２は，乱数ｋを生成する。コミットメント生成部１３は，乱数生成部１２が生成した乱数ｋよりコミットメントｗを生成する。コミットメント出力部１４は，コミットメント生成部１３が生成したコミットメントｗを出力する。レスポンス生成部１０は，乱数生成部１２の生成した乱数ｋとチャレンジ入力部９に入力されたチャレンジｃとハッシュ値算出部６が生成したハッシュ値Ｘ（Ｍ）よりレスポンスｒを生成する。

【０１２３】

乱数生成部１２で生成された乱数ｋは，レスポンス生成部１０でレスポンスｒの生成に利用された直後に破棄するなどして，同一の乱数ｋが異なるレスポンスｒの生成に重複して利用されないように注意しなければならない。

【０１２４】

以下，レスポンスｒの具体的な生成方法を，Ｓｃｈｎｏｒｒ認証，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証，Ｆｉａｔ－Ｓｈａｍｉｒ認証を例にとって説明しよう。

【０１２５】

有限群Ｇは，Ｓｃｈｎｏｒｒ認証の例では離散対数問題が困難とし，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証，Ｆｉａｔ－Ｓｈａｍｉｒ認証の例では零化域決定問題が困難であるとする。

【０１２６】

ハッシュ値算出部６が生成するハッシュ値Ｘ（Ｍ）と乱数生成部１２が生成する乱数ｋは，Ｓｃｈｎｏｒｒ認証の例ではｐ元体F_pの元であり，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証，Ｆｉａｔ－Ｓｈａｍｉｒ認証の例では有限群Ｇの元である。

【０１２７】

チャレンジ入力部９に入力されるチャレンジｃは，Ｓｃｈｎｏｒｒ認証，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証の例ではｐ元体F_pの元であり，Ｆｉａｔ－Ｓｈａｍｉｒ認証の例では固定長のビット列である。

【０１２８】

[Ｓｃｈｎｏｒｒ認証１の場合]

コミットメントｗは

【０１２９】

【数１８】

$$w=\pi(k.\ g)$$

として生成され，レスポンスｒは

【０１３０】

【数１９】

$$r=k-cX(M)$$

として生成される。

【０１３１】

[Ｓｃｈｎｏｒｒ認証２の場合]

コミットメントｗは

【０１３２】

【数２０】

$$w=k.\ g$$

として生成され，レスポンスｒは

【０１３３】

【数２１】

$$r=k-cX(M)$$

として生成される。

【０１３４】

[Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証１の場合]

コミットメントｗは

【０１３５】

【数２２】

$$w=\pi(p.\ k)$$

として生成され，レスポンスｒは

【０１３６】

【数２３】

$$r = k - c.\ X(M)$$

として生成される。

【０１３７】

[Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証２の場合]

コミットメントｗは

【０１３８】

【数２４】

$$w = p.\ k$$

として生成され，レスポンスｒは

【０１３９】

【数２５】

$$r = k - c.\ X(M)$$

として生成される。

【０１４０】

[Ｆｉａｔ－Ｓｈａｍｉｒ認証１の場合]

コミットメントｗは

【０１４１】

【数２６】

$$w = \pi\ (2.\ k)$$

として生成され，レスポンスｒは

【０１４２】

【数２７】

$$r = k - \Sigma\, c_i.\ X(M)_i$$

として生成される。ただし，

【０１４３】

【数２８】

$$c = (c_1, \ldots, c_n)$$

$$X(M) = (X(M)_1, \ldots, X(M)_n)$$

である。

【０１４４】

[Ｆｉａｔ－Ｓｈａｍｉｒ認証２の場合]

コミットメントｗは

【０１４５】

【数２９】

$$w = 2.\ k$$

として生成され，レスポンスｒは

【０１４６】

【数３０】

$$r = k - \Sigma c_i.\ X(M)_i$$

として生成される。ただし，

【０１４７】

【数３１】

$$c = (c1, \ldots, cn)$$

$$X(M) = (X(M)_1, \ldots, X(M)_n)$$

である。

【０１４８】

[実施例５：証明装置（擬３方向認証）]

本実施例では，実施例２ないし４と異なる秘密鍵処理部８を持つ証明装置の例を図６に基づいて説明している。図６は秘密鍵処理部８の詳細な構成図である。

【０１４９】

秘密鍵処理部８は，実施例４の証明装置における秘密鍵処理部８の構成と同様である。

【０１５０】

レスポンス生成部１０は，乱数生成部１２の生成した乱数ｋとコミットメント生成部１３の生成したコミットメントｗとチャレンジ入力部９に入力されたチャレンジｃとハッシュ値算出部６が生成したハッシュ値Ｘ（Ｍ）よりレスポンスｒを生成する。

【０１５１】

以下，レスポンスｒの具体的な生成方法を，ＤＳＡ認証を例にとって説明しよう。

【０１５２】

ここでは，有限群Ｇは離散対数問題が困難であるとする。また，ハッシュ値算出部６が生成するハッシュ値Ｘ（Ｍ）と乱数生成部１２が生成する乱数ｋとチャレンジ入力部９に入力されるチャレンジｃはｐ元体F_pの元である。

【０１５３】

[ＤＳＡ認証の場合]

コミットメントｗは

【０１５４】

【数３２】

$$w=\pi(k,\ g)$$

として生成され，レスポンスｒは

【０１５５】

【数３３】

$$r=(c-wX(M))/k$$

として生成される。

【０１５６】

ちなみに，レスポンスｒを

【０１５７】

【数３４】

$$r\leftarrow(w,\ r)$$

としてコミットメントｗとレスポンスｒの対（ｗ，ｒ）で置き換えると，レスポンスｒはチャレンジｃに対するＤＳＡ署名に他ならない。

【０１５８】

[実施例６：証明器発行装置]

本実施例では，固有値ｄを入力すると，固有値ｄに依存したハッシュ関数Ｘを備えた証明器（実施例２ないし５）を発行する装置を図７に基づいて説明する。

図７は証明器発行装置の構成図である。本実施例の証明器発行装置は，ハッシュ値生成装置（実施例１）において用いたものと等価な固有値入力部１と関数生成固有値記憶部３と値生成固有値算出部４を持つ。

【０１５９】

固有値入力部１は，ハッシュ関数Ｘの生成に要するパラメータである固有値ｄが入力される。関数生成固有値記憶部３は，値生成固有値の生成に要するパラメータである関数生成固有値ｓを保持する。値生成固有値算出部４は，関数生成固有値記憶部３に保持された関数生成固有値ｓと固有値入力部１に入力された固有値ｄより値生成固有値ｕを生成する。値生成固有値書込部１５は，値生成固有値算出部４が生成した値生成固有値ｕを証明器Ｔに書き込む。証明器発行部１６は，値生成固有値書込部１５が値生成固有値ｕを書き込んだ証明器Ｔを発行する。

【０１６０】

［実施例７：証明証型認証装置（２方向認証）］

本実施例では，証明器（実施例２，３）を用いた認証装置を図８と図９に基づいて説明する。図８は証明証型認証装置の構成図であり，図９は秘密鍵処理検証部の詳細な構成図である。

【０１６１】

証明証記憶部１７は，証明器に固有のハッシュ関数ＸによるメッセージＭのハッシュ値Ｘ（Ｍ）を秘密鍵と見なしたものと対をなす公開鍵ｙを証明する証明証Ｃを保持する。証明証検証部１８は，証明証記憶部１７に保持された証明証Ｃを検証し，検証に成功すれば，証明証Ｃが証明する公開鍵ｙを取得する。秘密鍵処理検証部１９は，証明証検証部１８が取得した公開鍵ｙに基づいて，証明器と対話をすることによって証明器の秘密鍵処理を検証する。

【０１６２】

この認証装置は，例えば，スマートカードリーダー内のＲＯＭに焼き付けられたプログラムとして実現しても良い。

【０１６３】

また，この認証装置は，例えば，アクセス制御が施されるアプリケーションプログラム内に検出が困難な様に埋入されたコードとして実現してもよい。アプリ

ケーションプログラムは実行時の要所要所でこのコードによって認証を行うことによって，ユーザーの利用資格を検証する。

【０１６４】

また，この認証装置は，例えば，アクセス制御が施されるデジタルコンテンツをレンダリングするアプリケーションプログラム内に検出が困難な様に埋入されたコードとして実現してもよい。保護されたコンテンツをレンダリングする場合，アプリケーションプログラムは実行時の要所要所でこのコードによって認証を行うことによって，ユーザーの利用資格を検証する。

【０１６５】

証明証検証部１８は，保持された証明証Ｃに基づいた処理を行ってもよい。例えば，証明証Ｃには有効期限情報等の利用条件が記載されており，利用条件が満たされない場合は証明証の検証に失敗するように構成してもよい。このように構成することで，証明証に有効期限情報等の利用条件を対応づけることが可能となる。

【０１６６】

また，例えば，証明証Ｃには認証の識別子が記載されており，その識別子が予見されるものと異なっている場合は証明証の検証に失敗するように構成してもよい。このように構成することで，証明証の正しさを保証する証明証発行者の署名鍵を固定したまま，複数の認証に対応することができる。

【０１６７】

秘密鍵処理検証部１９は，チャレンジ生成部２０とレスポンス検証部２１より構成される。

【０１６８】

チャレンジ生成部２０は，認証のための質疑情報であるチャレンジｃを生成する。レスポンス検証部２１は，証明証検証部１８より取得した公開鍵ｙとチャレンジ生成部２０が生成したチャレンジｃを用いて証明器のレスポンス出力部１１より取得したレスポンスｒを検証する。

【０１６９】

チャレンジ生成部２０はチャレンジｃをランダムに生成しても良いし，ハッシ

ュ関数ｈを用いて署名を生成したいメッセージｍのハッシュ値としてｃ　＝　ｈ（Ｍ）として生成しても良いし，復号したい暗号文Ｋを選びｃ　＝　Ｋとしてしても良いし，乱数ｋを生成し，復号したい暗号文Ｋに乱数ｋでブラインド効果を付与することによって生成しても良い。

【０１７０】

以下，レスポンスｒの検証方法を，様々なバリエーションを交えて説明しよう。

【０１７１】

証明証検証部１８より取得される公開鍵ｙは，ＤＳＡ署名，ＥｌＧａｍａｌ署名の変種，Ｎｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名，Ｓｃｈｎｏｒｒ署名，メッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の例では有限群Ｇの元であり，ＲＳＡの例では有限環Λの元である（ＲＳＡの場合，公開鍵ｙは実質的には整数と見なしている）。

【０１７２】

[ＤＳＡ署名の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１７３】

【数３５】

$$r_0=\pi((c/r_1)\cdot g+(r_0/r_1)\cdot y)$$

である。

【０１７４】

[変形ＥｌＧａｍａｌ署名１の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１７５】

【数３６】

$$r_0=\pi((r_1/c)\cdot g+(r_0/c)\cdot y)$$

である。

【０１７６】

[変形ＥｌＧａｍａｌ署名２の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１７７】

【数３７】

$$r_0=\pi((r_1/r_0)\cdot g+(c/r_0)\cdot y)$$

である。

【０１７８】

[変形Ｅｌ Ｇａｍａｌ署名３の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１７９】

【数３８】

$$r_0=\pi((r_0/r_1)\cdot g+(c/r_1)\cdot y)$$

である。

【０１８０】

[変形Ｅｌ Ｇａｍａｌ署名４の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１８１】

【数３９】

$$r_0=\pi((r_0/c)\cdot g+(r_1/c)\cdot y)$$

である。

【０１８２】

[変形Ｅｌ Ｇａｍａｌ署名５の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１８３】

【数４０】

$$r_0=\pi((c/r_0)\cdot g+(r_1/r_0)\cdot y)$$

である。

【０１８４】

[Ｎｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１８５】

【数４１】

$$c = r_0 + \pi\ (r_1.\ g + r_0.\ y)$$

である。

【０１８６】

[Ｓｃｈｎｏｒｒ署名]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１８７】

【数４２】

$$r_0 = h\ (c \mid \varepsilon\ (r_1.\ g + r_0.\ y))$$

である。

【０１８８】

[ＲＳＡの場合]

レスポンスｒの検証式は

【０１８９】

【数４３】

$$c = y.\ r$$

である。

【０１９０】

[メッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１９１】

【数４４】

$$c = r_0 + \pi\ (p.\ r_1 + y.\ r_0)$$

である。

【０１９２】

[Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の場合]

レスポンスｒ＝（r_0，r_1）の検証式は

【０１９３】

【数４５】

$$r_0 = h(c \mid \varepsilon(p.\ r_1 + y.\ r_0))$$

である。

【０１９４】

[実施例８：証明証型認証装置（２方向認証Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有）]

本実施例では，実施例７と異なる秘密鍵処理検証部１９を持つ認証装置の例を図１０に基づいて説明している。図１０は秘密鍵処理検証部１９の詳細な構成図である。

【０１９５】

秘密鍵処理検証部１９は，乱数生成部２２とチャレンジ生成部２０とレスポンス検証部２１より構成される。

【０１９６】

乱数生成部２２は乱数ｋを生成する。チャレンジ生成部２０は，乱数生成部２２の生成した乱数ｋより認証のための質疑情報であるチャレンジｃを生成する。レスポンス検証部２１は，証明証検証部１８より取得した公開鍵ｙと乱数生成部２２が生成した乱数ｋより証明器のレスポンス出力部１１から取得したレスポンスｒを検証する。

【０１９７】

以下，チャレンジｃの生成方法とレスポンスｒの検証式を，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有を例にとって具体的に説明しよう。

【０１９８】

証明証検証部１８より取得される公開鍵ｙおよび乱数生成部２２で生成される乱数ｋは有限群Ｇの元である。

【０１９９】

[Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有の場合]

チャレンジｃは

【０２００】

【数４６】

ｃ＝ｋ．ｇ

として生成される。レスポンスｒの検証式は

【０２０１】

【数４７】

ｋ．ｙ＝ｒ

である。

【０２０２】

［実施例９：証明証型認証装置（３方向認証）］

本実施例では，証明器（実施例４）を用いた認証装置を図８と図１１に基づいて説明する。図８は証明証型認証装置の構成図であり，図１１は秘密鍵処理検証部の詳細な構成図である。

【０２０３】

秘密鍵処理検証部１９がチャレンジ生成部２０とレスポンス検証部２１より構成されるのは実施例７と同様である。

【０２０４】

チャレンジ生成部２０は，認証のための質疑情報であるチャレンジｃをランダムに生成する。レスポンス検証部２１は，証明証検証部１８より取得した公開鍵ｙとチャレンジｃの生成に先立って証明器のコミットメント出力部１１より取得したコミットメントｗとチャレンジ生成部２２が生成したチャレンジｃを用いて証明器のレスポンス出力部１１より取得したレスポンスｒを検証する。

【０２０５】

以下，レスポンスｒの検証方法をＳｃｈｎｏｒｒ認証とＧｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証を例にとって具体的に説明しよう。

【０２０６】

証明証検証部１８より取得される公開鍵ｙは有限群Ｇの元である。

【０２０７】

［Ｓｃｈｎｏｒｒ認証の場合］

レスポンスｒの検証式は

【０２０８】

【数４８】

$$w=\pi(r.\ g+c.\ y))$$

である。

【０２０９】

[Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証の場合]

レスポンスｒの検証式は

【０２１０】

【数４９】

$$w=\pi(p.\ r+c.\ y))$$

である。

【０２１１】

[Ｆｉａｔ－Ｓｈａｍｉｒ認証の場合]

レスポンスｒの検証式は

【０２１２】

【数５０】

$$w=\pi(p.\ r+\Sigma c_i.\ y_i)$$

である。

【０２１３】

[実施例１０：証明証発行装置]

本実施例では，固有値ｄとメッセージＭを入力すると，ハッシュ値生成器（実施例１）を用いて，固有値ｄに対応づけて発行された証明器（実施例２ないし６）を用いた証明証型認証装置（実施例７ないし９）で利用される証明証Ｃを発行する装置を，図１０に基づいて説明する。図１０は証明証発行装置の構成図である。

【０２１４】

公開鍵算出部２３は，ハッシュ値生成器の生成したハッシュ値Ｘ（Ｍ）を秘密鍵と見なし，対をなす公開鍵ｙを算出する。証明証発行部２４は，公開鍵算出部２３が算出した公開鍵ｙを証明する証明証Ｃを発行する。

【０２１５】

証明証発行部２４は，メッセージMに認証の識別子ａｉｄを記載するように構成しても良い。このようにすることで異なる認証の識別子に対応するハッシュ値Ｘ（Ｍ）が確実に異なるものにすることができる。

【０２１６】

以下，公開鍵ｙの算出方法を具体的に説明する。

【０２１７】

[離散対数問題が困難な有限群Ｇの場合]

公開鍵ｙは

【０２１８】

【数５１】

$$y = X(M) . g$$

として算出される。

【０２１９】

[零化域決定問題が困難な有限群Ｇの場合１]

Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名，メッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名では，公開鍵ｙは

【０２２０】

【数５２】

$$y = p . X(M)$$

として算出される。

【０２２１】

Ｆｉａｔ－Ｓｈａｍｉｒ認証では，公開鍵ｙは

【０２２２】

【数５３】

$$y_i = 2 . X(M)_i$$

として算出される。ただし，

【０２２３】

【数５４】

$y=(y_1, \ldots, y_n)$

$X(M)=(X(M)_1, \ldots, X(M)_n)$

である。

【０２２４】

［零化域決定問題が困難な有限群Ｇの場合２］

ＲＳＡでは，公開鍵ｙは

【０２２５】

【数５５】

$$y=X(M)^{-1}$$

として算出される。

【０２２６】

［実施例１１：チケット型認証装置（チャレンジを変換）］

本実施例では，証明器（実施例２，３）を用いた認証装置を図１３と図１４に基づいて説明する。図１３はアクセスチケット型認証装置の構成図である。本実施例の認証装置は，証明証型認証装置（実施例７ないし９）と等価な秘密鍵処理検証部１９を持つ。図１４は秘密鍵処理変換部２７の詳細な構成図である。

【０２２７】

公開鍵記憶部２５は，公開鍵ｙを保持する。アクセスチケット記憶部２６は，公開鍵と対をなす秘密鍵とハッシュ値Ｘ（Ｍ）とから定まるアクセスチケットｔを保持する。秘密鍵処理検証部１９は，公開鍵記憶部２５が保持する公開鍵ｙに基づいて証明器と対話をすることによって証明器の秘密鍵処理を検証するが，この際，秘密鍵処理変換部２７が，アクセスチケット記憶部２６が保持するアクセスチケットｔを用いて証明器の秘密鍵処理を変換する。

【０２２８】

この認証装置は，例えば，スマートカードリーダー内のＲＯＭに焼き付けられたプログラムとして実現しても良い。

【０２２９】

また，この認証装置は，例えば，アクセス制御が施されるアプリケーションプログラム内に検出が困難な様に埋入されたコードとして実現してもよい。アプリ

ケーションプログラムは実行時の要所要所でこのコードによって認証を行うことによって，ユーザーの利用資格を検証する。

【０２３０】

秘密鍵処理検証部１９は，少なくともチャレンジ生成部２０を備え，秘密鍵処理部８は，少なくともチャレンジ入力部９を備える。

【０２３１】

秘密鍵処理変換部２７は，チャレンジ更新部２８より構成される。

【０２３２】

チャレンジ更新部２８は，チャレンジ生成部２０の生成したチャレンジｃをアクセスチケット記憶部２６が保持するアクセスチケットｔを用いて更新し，更新されたチャレンジｃをチャレンジ入力部９に入力する。

【０２３３】

以下，アクセスチケットｔの定義方法とチャレンジｃの更新方法を，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有と変形ＥｌＧａｍａｌ署名３とＳｃｈｎｏｒｒ認証とＲＳＡを例にとって説明しよう。

【０２３４】

秘密鍵ｘは，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有，変形ＥｌＧａｍａｌ署名３，Ｓｃｈｎｏｒｒ認証の例ではｐ元体F_pの元であり，ＲＳＡの例では有限環Λの元である（ＲＳＡの場合，対応するアクセスチケットｔは実質的には整数と見なしている）。

【０２３５】

[Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有の場合]

アクセスチケットｔの定義式は

【０２３６】

【数５６】

$$t = x / X(M)$$

であり，チャレンジｃは

【０２３７】

【数５７】

ｃ←ｔ．ｃ

として変換される。

【０２３８】

[変形Ｅｌ Ｇａｍａｌ署名３の場合]

アクセスチケットｔの定義式は

【０２３９】

【数５８】

ｔ＝ｘ／Ｘ（Ｍ）

であり，チャレンジｃは

【０２４０】

【数５９】

ｃ←ｃｔ

として変換される。

【０２４１】

[Ｓｃｈｎｏｒｒ認証の場合]

アクセスチケットｔの定義式は

【０２４２】

【数６０】

ｔ＝ｘ／Ｘ（Ｍ）

であり，チャレンジｃは

【０２４３】

【数６１】

ｃ←ｃｔ

として変換される。

【０２４４】

[ＲＳＡの場合]

アクセスチケットｔの定義式は

【０２４５】

【数６２】

ｔ＝ｘ／Ｘ（Ｍ）

であり，チャレンジｃは

【０２４６】

【数６３】

ｃ←ｔ．ｃ

として変換される。

【０２４７】

［実施例１２：チケット型認証装置（レスポンスを変換）］

本実施例では，実施例１１と異なる秘密鍵処理変換部２７を持つ認証装置の例を図１５に基づいて説明している。図１５は秘密鍵処理変換部２７の詳細な構成図である。

【０２４８】

秘密鍵処理部８は，少なくともレスポンス出力部１１を備える。

【０２４９】

秘密鍵処理変換部２７は，レスポンス更新部２９より構成される。

【０２５０】

レスポンス更新部２９は，レスポンス出力部１１の出力したレスポンスｒをアクセスチケット記憶部２６が保持するアクセスチケットｔを用いて更新する。

【０２５１】

以下，アクセスチケットｔの定義方法とレスポンスｒの具体的な更新方法を，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有と変形ＥｌＧａｍａｌ署名１と変形ＥｌＧａｍａｌ署名４と変形ＥｌＧａｍａｌ署名５とＲＳＡとＮｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名とメッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名を例にとって説明しよう。

【０２５２】

秘密鍵ｘは，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有，変形ＥｌＧａｍａｌ署名１，変形ＥｌＧａｍａｌ署名４，変形ＥｌＧａｍａｌ署名５の例ではｐ元体F_pの元であり，ＲＳＡの例では有限環Λの元である（ＲＳＡの場合，対応するアクセスチケットｔは実質的には整数と見なしている）。

【０２５３】

[Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有の場合]

アクセスチケットｔの定義式は

【０２５４】

【数６４】

$$t = x / X\ (M)$$

であり，レスポンスｒは

【０２５５】

【数６５】

$$r \leftarrow t \cdot r$$

として変換される。

【０２５６】

[変形ＥｌＧａｍａｌ署名１の場合]

アクセスチケットｔの定義式は

【０２５７】

【数６６】

$$t = x - X\ (M)$$

であり，レスポンス$r = (r_0, r_1)$は

【０２５８】

【数６７】

$$r_1 \leftarrow r_1 - r_0 t$$

として変換される。

【０２５９】

[変形ＥｌＧａｍａｌ署名４の場合]

アクセスチケットｔの定義式は

【０２６０】

【数６８】

$$t = x / X\ (M)$$

であり，レスポンス$r = (r_0, r_1)$は

【０２６１】

【数６９】

$$r_1 \leftarrow r_1 / t$$

として変換される。

【０２６２】

[変形Ｅｌ Ｇａｍａｌ署名５の場合]

アクセスチケットｔの定義式は

【０２６３】

【数７０】

$$t = x / X\ (M)$$

であり，レスポンスｒ＝（r_0，r_1）は

【０２６４】

【数７１】

$$r_1 \leftarrow r_1 / t$$

として変換される。

【０２６５】

[ＲＳＡの場合]

アクセスチケットｔの定義式は

【０２６６】

【数７２】

$$t = x / X\ (M)$$

であり，チャレンジｃは

【０２６７】

【数７３】

$$r \leftarrow t \cdot r$$

として変換される。

【０２６８】

[Ｎｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名の場合]

アクセスチケットｔの定義式は

【０２６９】

【数７４】

$$t = x - X(M)$$

であり，レスポンス$r = (r_0, r_1)$は

【０２７０】

【数７５】

$$r_1 \leftarrow r_1 - r_0 t$$

として変換される。

【０２７１】

[メッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名の場合]

アクセスチケットｔの定義式は

【０２７２】

【数７６】

$$t = x - X(M)$$

であり，レスポンス$r = (r_0, r_1)$は

【０２７３】

【数７７】

$$r_1 \leftarrow r_1 - r_0 \cdot t$$

として変換される。

【０２７４】

[実施例１３：チケット型認証装置（チャレンジでレスポンスを変換）]

本実施例では，実施例１１ないし１２と異なる秘密鍵処理変換部２７を持つ認証装置の例を図１６に基づいて説明している。図１６は秘密鍵処理変換部２７の詳細な構成図である。

【０２７５】

秘密鍵処理検証部１９は，少なくともチャレンジ生成部２０を備え，秘密鍵処理部８は，少なくともレスポンス出力部１１を備える。

【０２７６】

秘密鍵処理変換部２７は，レスポンス更新部２９より構成される。

【０２７７】

レスポンス更新部２９は，レスポンス出力部１１の出力したレスポンスｒをアクセスチケット記憶部２６が保持するアクセスチケットｔとチャレンジ生成部２０が生成したチャレンジｃを用いて更新する。

【０２７８】

以下，アクセスチケットｔの定義方法とレスポンスｒの更新方法を，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有，ＲＳＡ，変形ＥｌＧａｍａｌ署名２，Ｓｃｈｎｏｒｒ認証，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証を例にとって説明しよう。

【０２７９】

秘密鍵ｘは，Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有，変形ＥｌＧａｍａｌ署名１，変形ＥｌＧａｍａｌ署名４，変形ＥｌＧａｍａｌ署名５の例ではｐ元体F_pの元であり，ＲＳＡの例では有限環Λの元である（ＲＳＡの場合，対応するアクセスチケットｔは実質的には整数と見なしている）。

【０２８０】

[Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵共有の場合]

アクセスチケットｔの定義式は

【０２８１】

【数７８】

$$t = x - X(M)$$

であり，レスポンスｒは

【０２８２】

【数７９】

$$r \leftarrow r + t.\ c$$

として変換される。

【０２８３】

[ＲＳＡの場合]

アクセスチケットｔの定義式は

【０２８４】

【数８０】

$$t = x - X(M)$$

であり，レスポンスｒは

【０２８５】

【数８１】

$$r \leftarrow r + t.c$$

として変換される。

【０２８６】

[Ｓｃｈｎｏｒｒ認証の場合]

アクセスチケットｔの定義式は

【０２８７】

【数８２】

$$t = x - X(M)$$

であり，レスポンスｒは

【０２８８】

【数８３】

$$r \leftarrow r + ct$$

として変換される。

【０２８９】

[Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証の場合]

アクセスチケットｔの定義式は

【０２９０】

【数８４】

$$t = x - X(M)$$

であり，レスポンスｒは

【０２９１】

【数８５】

$$r \leftarrow r + c.t$$

として変換される。

【０２９２】

[実施例１４：チケット型認証装置（更新済みチャレンジでレスポンスを変換）]

本実施例では，実施例１１ないし１３と異なる秘密鍵処理変換部２７を持つ認証装置の例を図１７に基づいて説明している。図１７は秘密鍵処理変換部２７の詳細な構成図である。

【０２９３】

秘密鍵処理検証部１９は，少なくともチャレンジ生成部２０を備え，秘密鍵処理部８は，少なくともコミットメント出力部１１とチャレンジ入力部９とレスポンス出力部１１を備える。

【０２９４】

秘密鍵処理変換部２７は，チャレンジ更新部２８とレスポンス更新部２９より構成される。

【０２９５】

チャレンジ更新部２８は，チャレンジ生成部２０が生成したチャレンジｃをコミットメント出力部１４より取得したコミットメントｗを用いて更新し，チャレンジ入力部９に更新されたチャレンジｃを入力する。

【０２９６】

レスポンス更新部２９は，レスポンス出力部１１の出力したレスポンスｒをアクセスチケット記憶部２６が保持するアクセスチケットｔとチャレンジ更新部２８が更新したチャレンジｃを用いて更新する。

【０２９７】

以下，アクセスチケットｔの定義方法とチャレンジｃおよびレスポンスｒの更新方法を，Ｓｃｈｎｏｒｒ認証とＧｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証を例にとって説明しよう。

【０２９８】

秘密鍵ｘは，Ｓｃｈｎｏｒｒ認証の例ではｐ元体F_pの元であり，Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証の例では有限群Ｇの元である。

【０２９９】

［Ｓｃｈｎｏｒｒ認証１からＮｙｂｅｒｇ－Ｒｕｅｐｐｅｌ署名への変換の場合］

アクセスチケットｔの定義式は

【０３００】

【数８６】

$$t = x - X(M)$$

であり，チャレンジｃは

【０３０１】

【数８７】

$$c \leftarrow c - w$$

として変換され，レスポンスｒは

【０３０２】

【数８８】

$$r \leftarrow r + ct$$

として変換される。

【０３０３】

［Ｓｃｈｎｏｒｒ認証２からＳｃｈｎｏｒｒ署名への変換の場合］

アクセスチケットｔの定義式は

【０３０４】

【数８９】

$$t = x - X(M)$$

であり，チャレンジｃは

【０３０５】

【数９０】

$$c \leftarrow \eta(c \mid \varepsilon(w))$$

として変換され，レスポンスｒは

【０３０６】

【数９１】

$$r \leftarrow r + ct$$

として変換される。

【０３０７】

[Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証１からメッセージ復元型Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名への変換の場合]

アクセスチケットｔの定義式は

【０３０８】

【数９２】

$$t = x - X(M)$$

であり，チャレンジｃは

【０３０９】

【数９３】

$$c \leftarrow c - w$$

として変換され，レスポンスｒは

【０３１０】

【数９４】

$$r \leftarrow r + ct$$

として変換される。

【０３１１】

[Ｇｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ認証２からＧｕｉｌｌｏｕ－Ｑｕｉｓｑｕａｔｅｒ署名への変換の場合]

アクセスチケットｔの定義式は

【０３１２】

【数９５】

$$t = x - X(M)$$

であり，チャレンジｃは

【０３１３】

【数９６】

$$c \leftarrow \eta(c \mid \varepsilon(w))$$

として変換され，レスポンスｒは

【０３１４】

【数９７】

$$r \leftarrow r + ct$$

として変換される。

【０３１５】

［実施例１５：チケット型認証装置（コミットメントでチャレンジを変換）］

本実施例では，実施例１１ないし１４と異なる秘密鍵処理変換部２７を持つ認証装置の例を図１８に基づいて説明している。図１８は秘密鍵処理変換部２７の詳細な構成図である。

【０３１６】

秘密鍵処理検証部１９は，少なくともチャレンジ生成部２０を備え，秘密鍵処理部８は，少なくともコミットメント出力部１１とチャレンジ入力部９とレスポンス出力部１１を備える。

【０３１７】

秘密鍵処理変換部２７は，チャレンジ更新部２８とレスポンス更新部２９より構成される。

【０３１８】

チャレンジ更新部２８は，チャレンジ生成部２０が生成したチャレンジｃをアクセスチケット記憶部２６が保持するアクセスチケットｔとコミットメント出力部１４より取得したコミットメントｗを用いて更新し，チャレンジ入力部９に更新されたチャレンジｃを入力する。

【０３１９】

レスポンス更新部２９は，レスポンス出力部１１の出力したレスポンスｒをコミットメント出力部１４より取得したコミットメントｗを用いて更新する。

【０３２０】

以下，アクセスチケットｔの定義方法とチャレンジｃおよびレスポンスｒの更新方法を，ＤＳＡ認証を例にとって説明しよう。

【０３２１】

秘密鍵ｘはｐ元体F_pの元である。

【０３２２】

[ＤＳＡ認証からＤＳＡ署名への変換の場合]

アクセスチケットｔの定義式は

【０３２３】

【数９８】

$$t = x - X(M)$$

であり，チャレンジｃは

【０３２４】

【数９９】

$$c \leftarrow c + wt$$

として変換され，レスポンスｒは

【０３２５】

【数１００】

$$r \leftarrow (w, r)$$

として変換される。

【０３２６】

[実施例１６：チケット発行装置]

本実施例では，固有値ｄとメッセージＭを入力すると，ハッシュ値生成器（実施例１）を用いて，固有値ｄに対応づけて発行された証明器（実施例２ないし６）を用いたアクセスチケット型認証装置（実施例１１ないし１５）で利用されるアクセスチケットｔを発行する装置を，図１９に基づいて説明する。図１９はアクセスチケット発行装置の構成図である。

【０３２７】

アクセスチケット算出部３１は，ハッシュ値生成器の生成したハッシュ値Ｘ（Ｍ）と秘密鍵記憶部３０に保持された秘密鍵ｘよりアクセスチケットｔを算出する。アクセスチケット発行部３２は，アクセスチケット算出部３１が算出したアクセスチケットｔを発行する。

【０３２８】

メッセージＭに認証の識別子ａｉｄを含ませるように構成しても良い。このよ

うにすることで異なる認証の識別子に対応するハッシュ値Ｘ（Ｍ）が確実に異なるものにすることができる。

【０３２９】

以下，アクセスチケットｔの算出方法を具体的に説明する。

【０３３０】

[秘密鍵がｐ元体F_pの元の場合]

アクセスチケットｔは

【０３３１】

【数１０１】

$$t = x - X(M)$$

あるいは

$$t = x / X(M)$$

として算出される。

【０３３２】

[秘密鍵が有限群Ｇの元の場合]

アクセスチケットｔは

【０３３３】

【数１０２】

$$t = x - X(M)$$

として算出される。

【０３３４】

[秘密鍵が有限環Λの元の場合]

アクセスチケットｔは

【０３３５】

【数１０３】

$$t = x - X(M)$$

あるいは

$$t = x / X(M)$$

として算出される。

【０３３６】

[実施例１７：認証システム]

本実施例では，証明証型認証装置（実施例７ないし９）あるいはアクセスチケット型認証装置（実施例１１ないし１５）を用いた認証システムを図２０に基づいて説明する。図２０は認証システムの概念構成図である。

【０３３７】

資格発行者（センター）３３は，ケーパビリティの発行器（実施例１０あるいは１６）３４と，資格付与者に対応づけられた固有値ｄを保持している。

【０３３８】

資格付与者（ユーザー）３５は，証明器発行装置（実施例６）によって固有値ｄに対応づけられて発行された秘密ハッシュ関数Ｈを内部に持つ証明器（実施例２ないし５）３６を保持している。

【０３３９】

資格発行者３３が資格付与者３５に資格を付与する場合，メッセージＭに対応づけて資格の保持を表象するケーパビリティχを，ケーパビリティの発行装置を用いて資格付与者３５に発行する。

【０３４０】

資格付与者３５が資格検証者３７に資格が付与されたことを証明する場合，ケーパビリティχと証明装置を含む認証装置を用いる。

【０３４１】

証明証型認証システムの場合，認証のケーパビリティは，トークン内の秘密ハッシュ値に対応する公開鍵と，認証の識別子を含み，資格検証者が信頼できる者の署名が施された証明証としてユーザーに与えられる。証明証の発行者は，証明証発行装置（実施例１０）を用いて固有値ｄに対応するトークンの保持者たるユーザーにメッセージＭに対応づけて証明証を発行することができる。固有値としては，例えば，トークン識別子ｔｉｄを，メッセージＭとしては，例えば，認証識別子ａｉｄを選ぶことができる。

【０３４２】

アクセスチケット型認証システムの場合，認証のケーパビリティは，トークン

内の秘密ハッシュ値と認証の識別子に対応する秘密鍵から計算される量であるアクセスチケットとしてユーザーに与えられる。アクセスチケットの発行者は，アクセスチケット発行装置（実施例１６）を用いて固有値ｄに対応するトークンの保持者たるユーザーにメッセージMに対応づけてアクセスチケットを発行することができる。固有値としては，例えば，トークン識別子ｔｉｄを，メッセージMとしては，例えば，認証識別子ａｉｄを選ぶことができる。

【０３４３】

【発明の効果】

以上，説明してきたように，本発明の一方向性関数生成技術およびそれに基づいた認証手法では，利用者が単一のトークンを備えることによって，単純な原理で多彩なケーパビリティを利用可能となる。

【図面の簡単な説明】

【図１】　ハッシュ値生成装置の構成を示す図である。

【図２】　証明装置の構成を示す図である。

【図３】　秘密鍵処理部の詳細な構成を示す図である。

【図４】　秘密鍵処理部の詳細な構成を示す図である。

【図５】　秘密鍵処理部の詳細な構成を示す図である。

【図６】　秘密鍵処理部の詳細な構成を示す図である。

【図７】　証明器発行装置の構成を示す図である。

【図８】　証明証型認証装置の構成を示す図である。

【図９】　秘密鍵処理検証部の詳細な構成を示す図である。

【図１０】　秘密鍵処理検証部の詳細な構成を示す図である。

【図１１】　秘密鍵処理検証部の詳細な構成を示す図である。

【図１２】　証明証発行装置の構成を示す図である。

【図１３】　アクセスチケット型認証装置の構成を示す図である。

【図１４】　秘密鍵処理変換部の詳細な構成を示す図である。

【図１５】　秘密鍵処理変換部の詳細な構成を示す図である。

【図１６】　秘密鍵処理変換部の詳細な構成を示す図である。

【図１７】　秘密鍵処理変換部の詳細な構成を示す図である。

【図１８】　秘密鍵処理変換部の詳細な構成を示す図である。

【図１９】　アクセスチケット発行装置の構成を示す図である。

【図２０】　認証システムの概念構成を示す図である。

【符号の説明】

１　固有値入力部

２　メッセージ入力部

３　関数生成固有値記憶部

４　値生成固有値算出部

５　ハッシュ値算出部

６　ハッシュ値出力部

７　値生成固有値記憶部

８　秘密鍵処理部

９　チャレンジ入力部

１０　レスポンス生成部

１１　レスポンス出力部

１２　乱数生成部

１３　コミットメント生成部

１４　コミットメント出力部

１５　値生成固有値書込部

１６　証明器発行部

１７　証明証記憶部

１８　証明証検証部

１９　秘密鍵処理検証部

２０　チャレンジ生成部

２１　レスポンス検証部

２２　乱数生成部

２３　公開鍵算出部

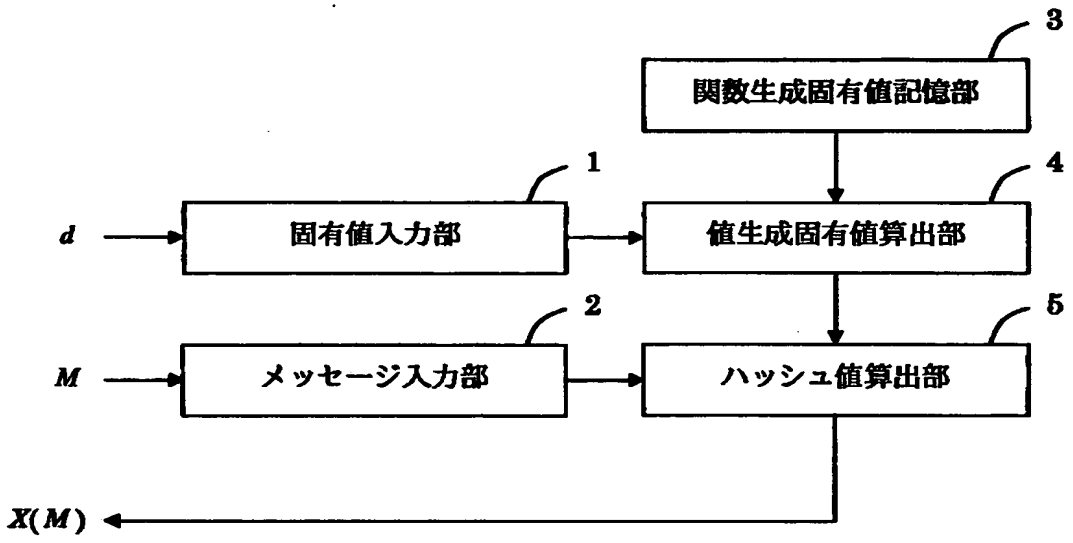
２４　証明証発行部

２５　公開鍵記憶部

２６　アクセスチケット記憶部
２７　秘密鍵処理変換部
２８　チャレンジ更新部
２９　レスポンス更新部
３０　秘密鍵記憶部
３１　アクセスチケット算出部
３２　アクセスチケット発行部
３３　資格発行者
３４　ケーパビリティ発行器
３５　資格付与者
３６　証明器
３７　資格検証者

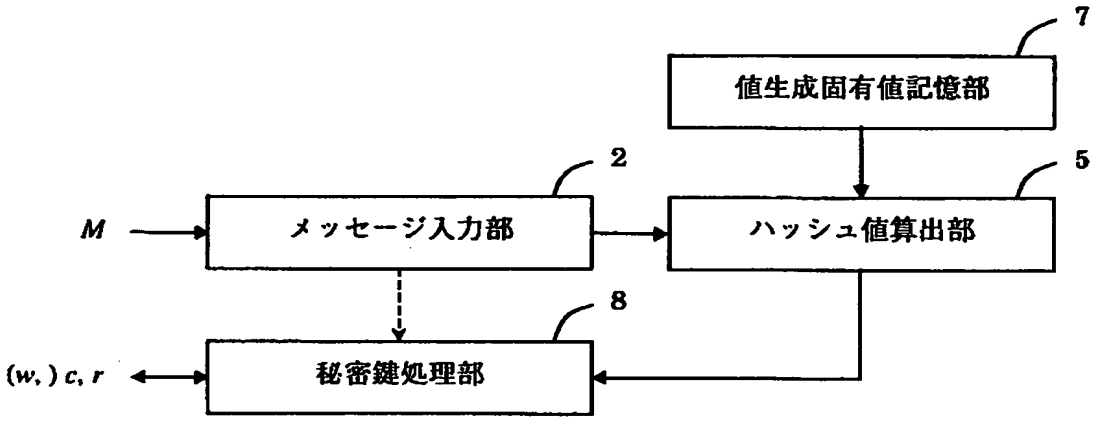
【書類名】　　図面

【図１】



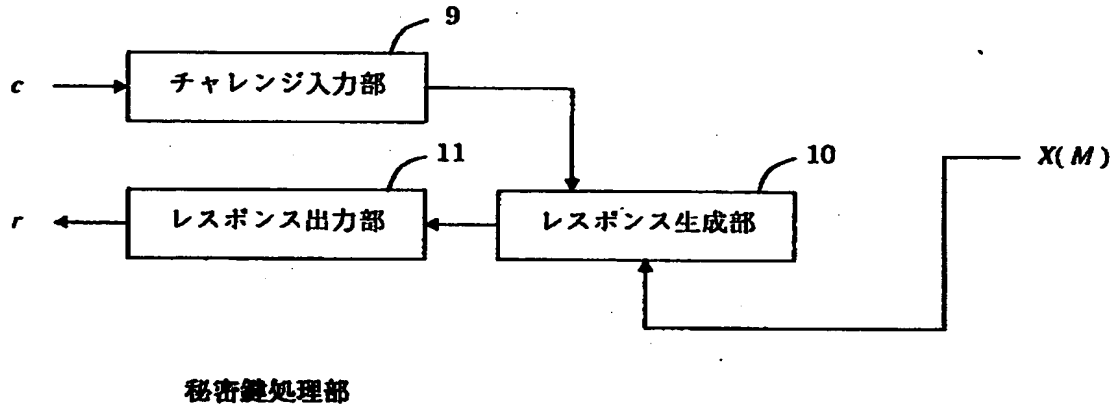
ハッシュ値生成装置

【図２】

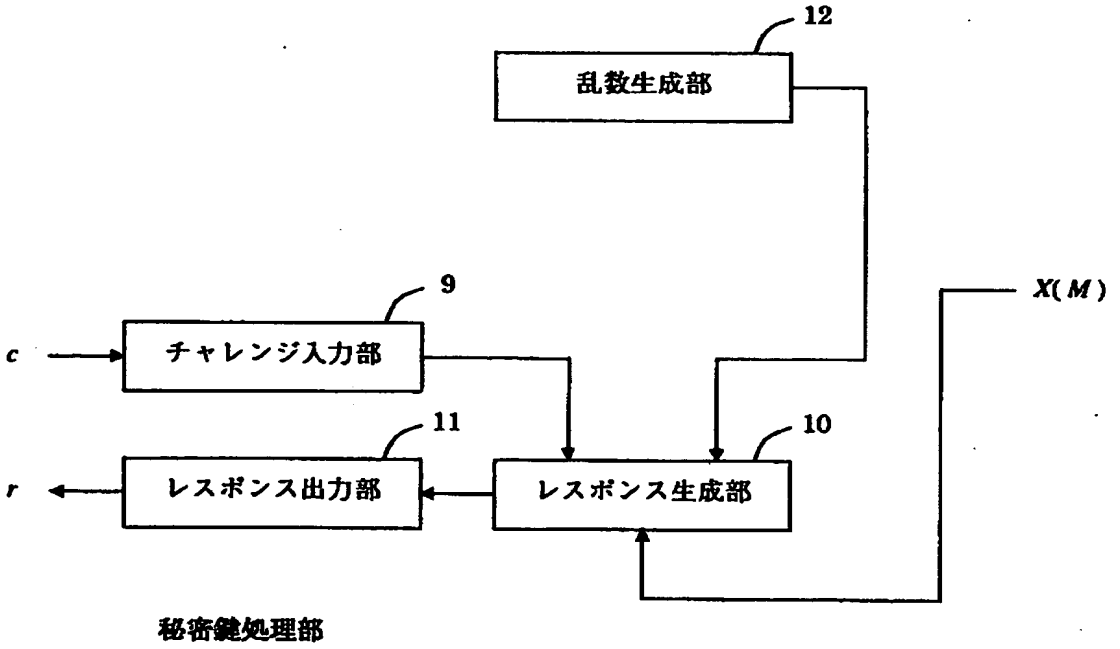


証明装置

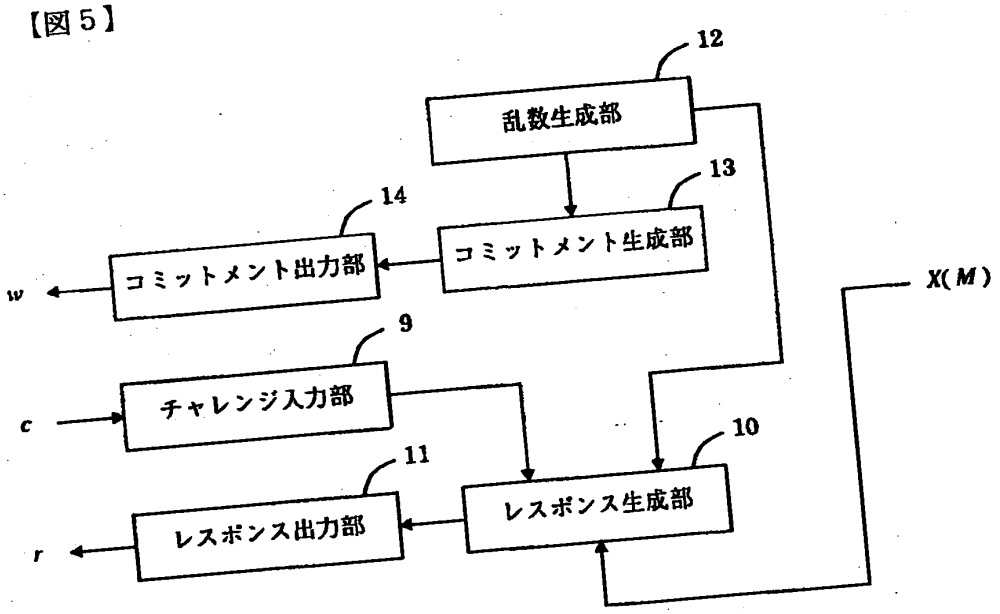
【図３】


9
c
チャレンジ入力部
11
10
X(M)
r
レスポンス出力部
レスポンス生成部
秘密鍵処理部

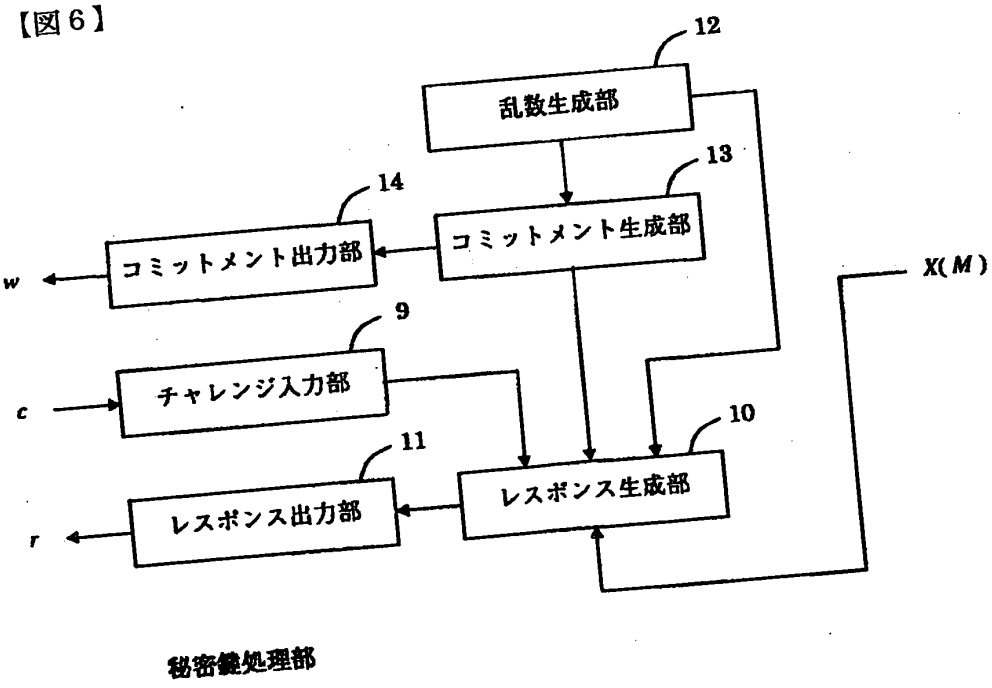
【図４】


12
乱数生成部
9
X(M)
c
チャレンジ入力部
11
10
r
レスポンス出力部
レスポンス生成部
秘密鍵処理部

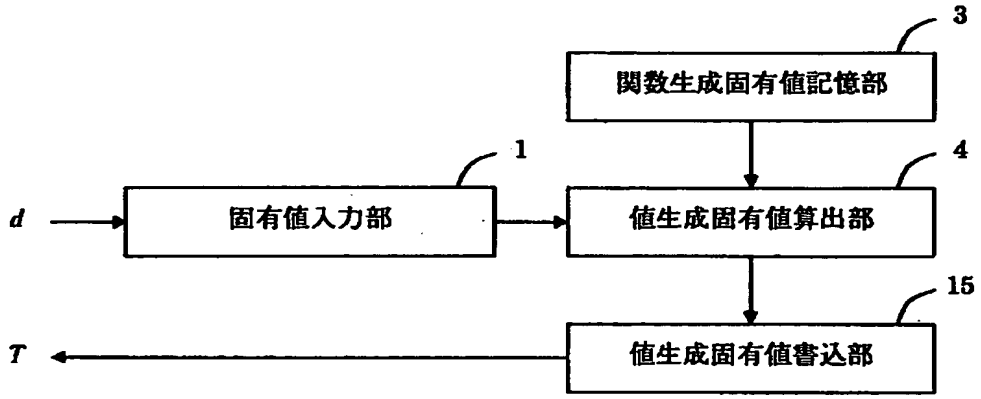
【図５】


12
乱数生成部
13
14
コミットメント生成部
コミットメント出力部
w
X(M)
9
チャレンジ入力部
c
10
11
レスポンス出力部
レスポンス生成部
r
秘密鍵処理部

【図６】

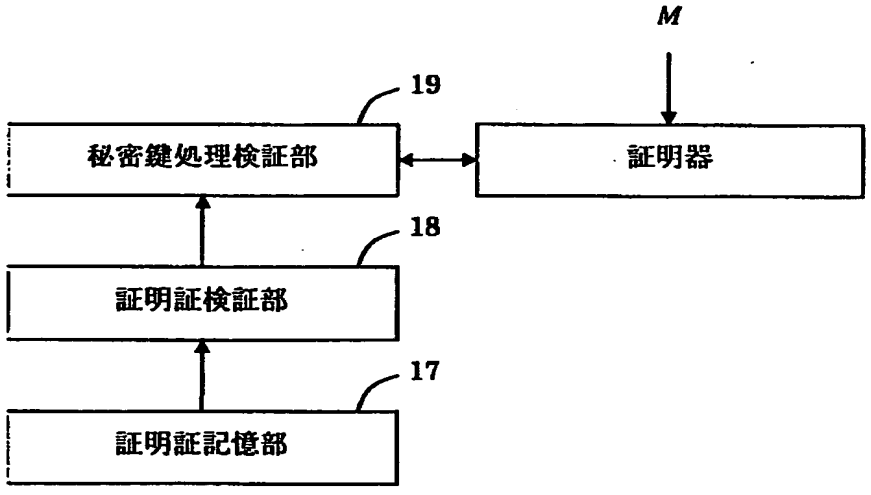

12
乱数生成部
13
14
コミットメント生成部
コミットメント出力部
w
X(M)
9
チャレンジ入力部
c
10
11
レスポンス出力部
レスポンス生成部
r
秘密鍵処理部

【図７】



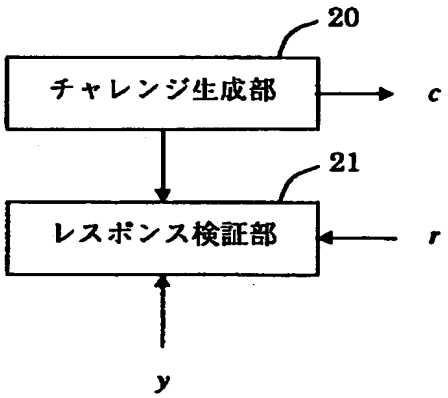
証明器発行装置

【図８】



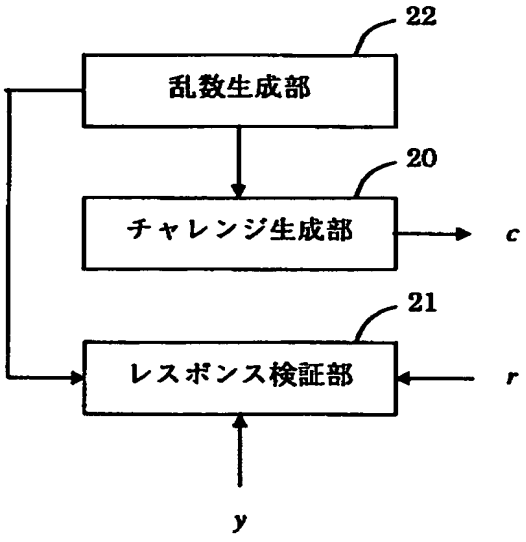
証明証型認証装置

【図９】



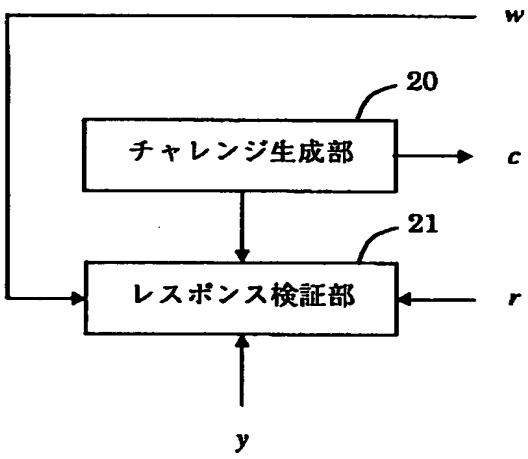
秘密鍵処理検証部

【図１０】



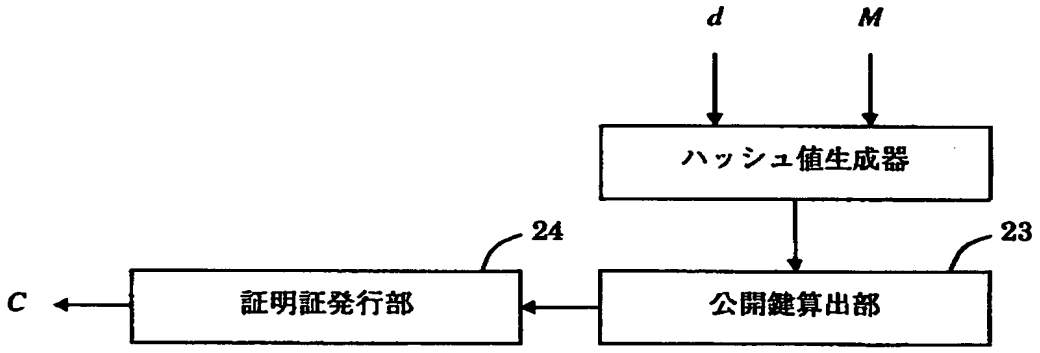
秘密鍵処理検証部

【図１１】



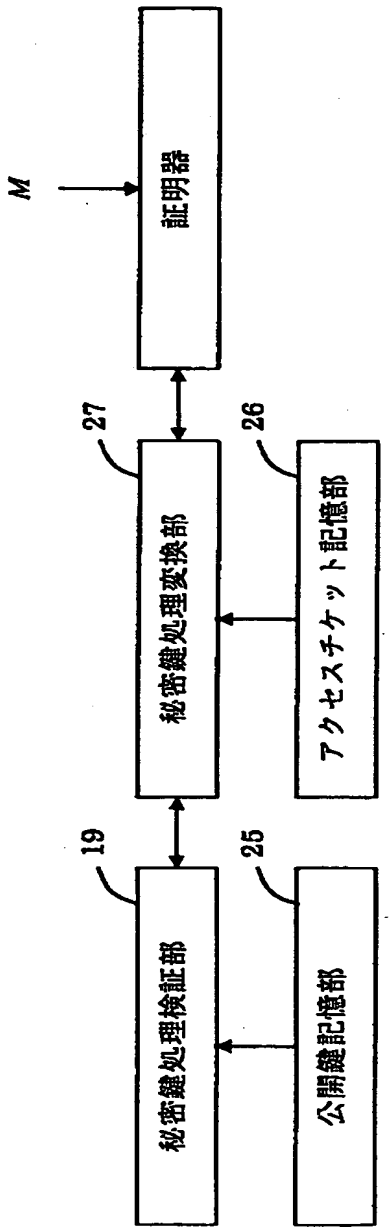
秘密鍵処理検証部

【図１２】



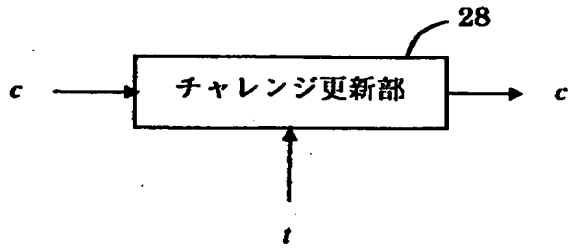
証明証発行装置

【図１３】



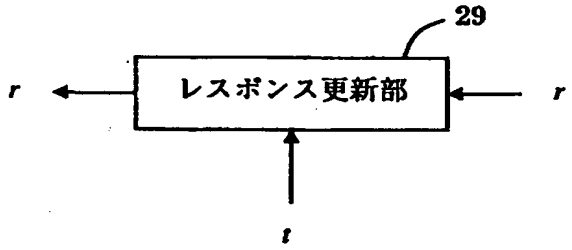
アクセスチケット型認証装置

【図１４】



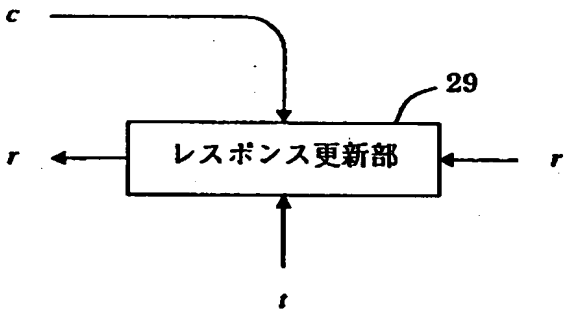
秘密鍵処理変換部

【図１５】



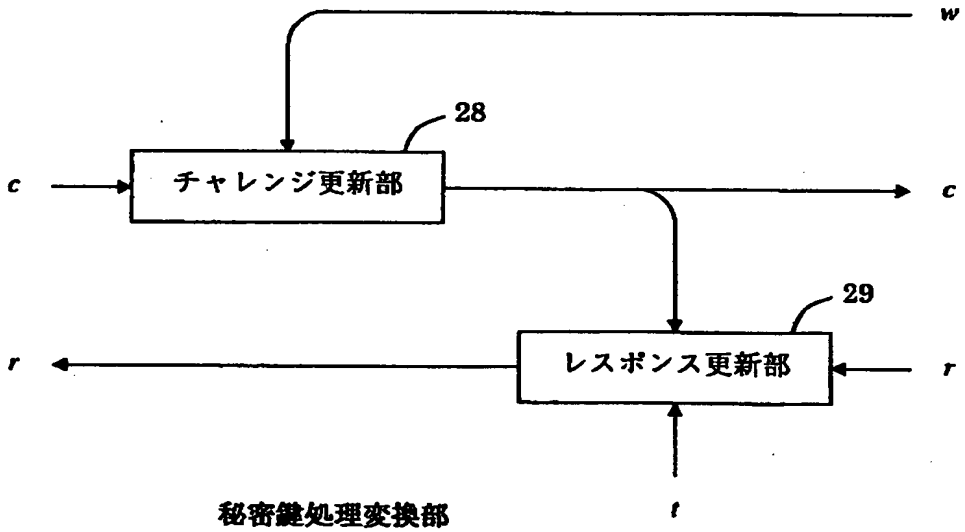
秘密鍵処理変換部

【図１６】



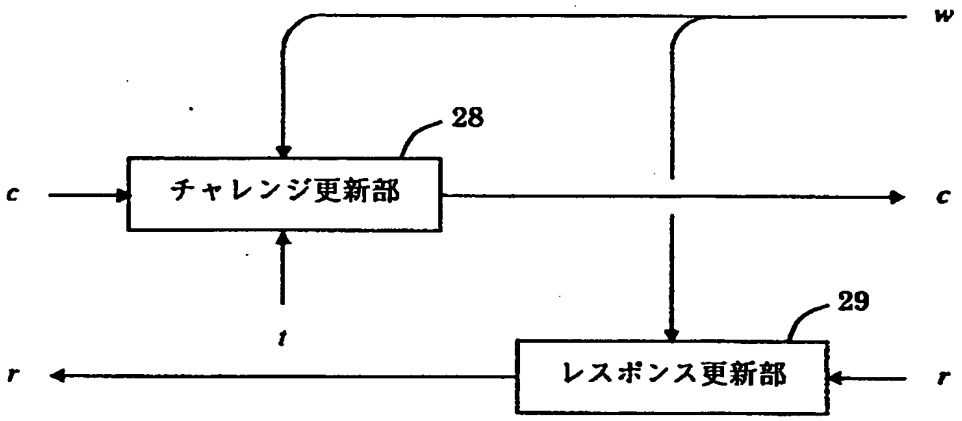
秘密鍵処理変換部

【図１７】



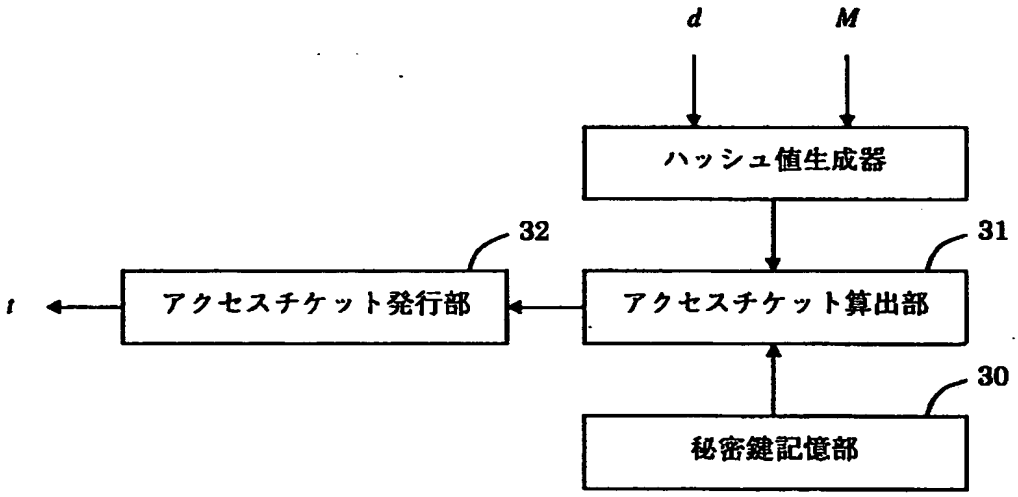
秘密鍵処理変換部

【図１８】



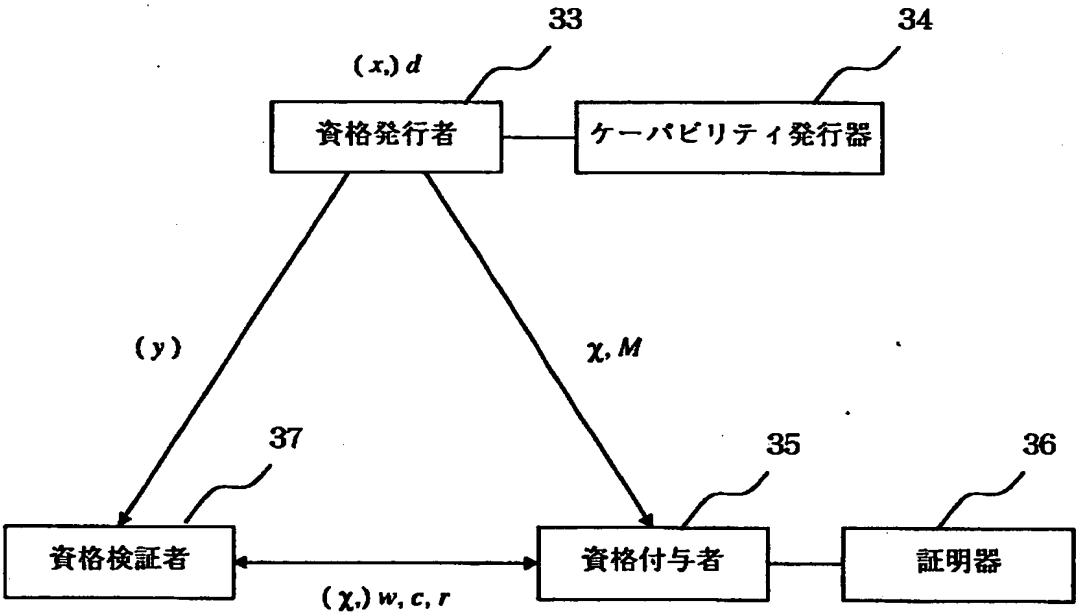
秘密鍵処理変換部

【図１９】



アクセスチケット発行装置

【図２０】



認証システム

【書類名】　要約書

【要約】

【課題】　トークンごとにハッシュ関数の秘密固有値を保管管理する必要なしに，かつセンターの組織的な秘密情報が暴露されるおそれなしに，トークンにハッシュ関数を付与する。

【解決手段】　固有値入力部１には，ハッシュ関数Ｘの生成に要するパラメータである固有値ｄが入力される。メッセージ入力部２には，ハッシュ値を求めたいメッセージＭが入力される。関数生成固有値記憶部３は，値生成固有値の生成に要するパラメータである関数生成固有値ｓを保持する。値生成固有値算出部４は，関数生成固有値ｓと固有値ｄより値生成固有値ｕを生成する。ハッシュ値算出部５は，値生成固有値ｕとメッセージＭよりハッシュ関数Ｈを用いてハッシュ値Ｘ（Ｍ）を生成する。ハッシュ値出力部６は，ハッシュ値算出部５が生成したハッシュ値Ｘ（Ｍ）を出力する。

【選択図】　図１

出　願　人　履　歴　情　報

識別番号　　〔０００００５４９６〕

１．変更年月日　　１９９６年　５月２９日
［変更理由］　住所変更
住　所　東京都港区赤坂二丁目１７番２２号
氏　名　富士ゼロックス株式会社